安全可以复制——向世界500强企业学习Linux安全管理与运维之道

51CTO真心推荐：献给工作在企业信息安全和Linux安全运维领域工作者的实用宝典。

李洋 编著

# 防线

## 企业Linux安全运维理念和实战

“一本真正适合中国企业的Linux安全管理与运维书籍”

“站在巨人肩膀上学习企业信息安全建设和Linux安全运维”

- 企业安全架构设计和原则
- 企业安全技术实战演练
- 企业安全标准和审计指导
- Linux安全管理和运维实现
- Linux开源安全运维工具和应用

清华大学出版社

# 数字版权声明

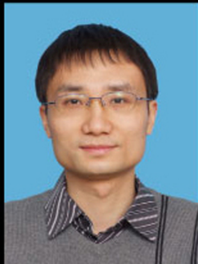
图灵社区的电子书没有采用专有客户端，您可以在任意设备上，用自己喜欢的浏览器和PDF阅读器进行阅读。

但您购买的电子书仅供您个人使用，未经授权，不得进行传播。

我们愿意相信读者具有这样的良知和觉悟，与我们共同保护知识产权。

如果购买者有侵权行为，我们可能对该用户实施包括但不限于关闭该帐号等维权措施，并可能追究法律责任。

■作者简介



**李洋** 博士

现任某信息安全公司CTO，历任金融公司信息安全顾问、电信运营商和互联网企业信息安全研究员/项目经理。

十余年来一直从事信息安全和IT架构领域的技术研发和管理工作，曾主持和参与多项国家重点项目，并主导多个电信网络、互联网网络、金融企业网络的IT架构设计、信息安全系统设计和研发工作。

具有丰富的企业信息安全规划、架构设计、建设和管理经验，擅长企业信息安全解决方案提供及实施、IT架构设计、网络和系统应用/管理/安全及操作系统内核的研发。

曾在IEEE、ACM、51CTO、《计算机世界》、《网管员世界》等国内外知名媒体和期刊上发表SCI/EI学术论文和各类技术文章百余篇，拥有个人专著6部，国家专利4项。

个人博客（2011年十大IT博客）
patterson.blog.51cto.com

# 防线

## 企业Linux安全运维理念和实战

李 洋 编著

清华大学出版社

北 京

# 前 言

## 本书的写作思路

### 人的正确思想是从哪里来的？——什么是安全（What）

“人的正确思想是从哪里来的？是从天上掉下来的吗？不是。是自己头脑里固有的吗？不是。人的正确思想，只能从社会实践中来，只能从生产斗争、阶级斗争和科学实验这三项实践中来。”毛泽东同志早在 1963 年 5 月就英明地作出过上述论断。

在写这本书之前，我就一直在思考这个问题，什么是安全？安全如何定义？安全的定义来自于哪里？……种种疑问，一直陪伴着我。进入信息安全行业已经 10 多个年头了，从技术研发，学术研究到目前的企业信息安全规划、管理、决策，大大小小的项目，层出不穷的安全威胁，琳琅满目的安全产品，日新月异的安全市场，如雨后春笋般出现的安全新名词，都使我更加坚定地意识到，在这本书里，必须把安全这个问题讲透彻，讲明白。在市面上很多书籍里面，包括操作系统安全、安全原理方面的书籍，以及我以前的几部著作中，都忽略了这个问题，这个问题没有阐述和介绍清楚，会直接影响到企业的信息安全工作的定位、实施和成效。

因此，在这本书的前面章节中，笔者结合目前业界最新的研究成果和自己 10 余年的信息安全从业经验，并根据著名信息安全领域专家方滨兴院士科学、系统定义的信息安全概念，作了有针对性地扩展和延伸，给出了信息安全定义、企业信息安全框架以及企业信息安全实施和建设的主要工作思路。有了这个前提，在后面章节的介绍中就有了一个科学的基调和前提，也便于向读者介绍 500 强企业使用开源 Linux 系统建设信息安全的思路和具体方法。

### 打破砂锅问到底——500 强企业为什么要关注安全（Why）

本书的笔者有多年的世界 500 强企业的信息安全管理工作经验，深谙 500 强企业信息安全建设、规划、实施和管理的细节、难点和重点问题。并且，笔者认为，世界 500 强企业对于信息安全工作的重视程度，以及信息安全在建设、规划、实施和管理等方面都有其所长，可以为其他中小型和大型企业所借鉴和参照。基于这个目的，本书以笔者在 500 强企业中使用企业级开源操作系统 Linux 在信息安全中的部署和使用方法为切入点，来介绍如何做好信息安全工作。

言归正传，500 强企业为什么要关注安全呢？原因是多方面的。首先，500 强企业有自己的服务器、操作系统，也有自己的系统，这些系统无论是为内部员工服务的（比如 OA 系统、电子邮件系统等），还是为外部客户服务的（比如 Web 发布系统、线上业务系统等），都会面临多方面的安全威胁，包括内部员工的恶意破坏、数据泄露、误操作等，以及外部不法用户的非授权访问、拒绝服务攻击、垃圾邮件、SQL 注入等，这些都需要进行安全方面的全面分析、研究、部署等，才能

将这些威胁对于企业的危害降到最低；其次，作为大型企业来说，需要从体制上来健全信息安全工作，包括规划、实施、流程制定、人员培训等，所以要特别关注信息安全。实践证明，越是规模大的企业，越容易出现安全问题，如果不居安思危，早作准备，就很难在现代企业中立于不败之地，也很难幸免于我们最近经常听到的密码泄露事件、客户资料泄密事件等。一旦遇到这些问题，企业的形象将会不保，企业的品牌和对于用户建立的信任也会受到很大影响；最后，国家、政府和行业也对不同的企业提出了很多信息安全方面的管制要求，主要是满足合规、生产安全等方面的要求，比如著名的国家等保、证监会/银监会的要求、支付行业的 PCI/DSS、美国的 SOX 法案等，都对企业尤其是大型企业提出了非常好的安全要求。

## 实践出真知——500 强企业如何做好信息安全建设（How）

明白了什么是安全，为什么要关注安全，那么最重要的一步就是如何来做安全了，也就是如何通过技术和管理手段来保证企业的安全，防止安全事件或者事故的发生，树立企业形象，保证企业正常运行。

根据笔者在 500 强企业的多年信息安全工作经验，在本书中，将使用企业级 Linux 进行信息安全建设工作的具体实施方法分为以下 5 个阶段：

1. 认知阶段；
2. 梳理阶段；
3. 实施阶段；
4. 运维阶段；
5. 工具应用阶段。

这 5 个阶段直接对应到本书的 5 个技术部分。会系统、全面地向读者介绍 500 强企业作好信息安全工作的方方面面。

# 读者群

本书是一本面向企业的基于操作系统平台进行信息安全建设的实践书籍，是围绕着什么是、为什么和怎么样构建信息安全来进行介绍的。因此，这本书并不需要读者具有高深的计算机科学与技术或者信息安全的基础理论知识，这本书的主要读者是怀有一定的工作目标并对信息安全有一定兴趣的工程师或者信息安全从业人员。

笔者也非常希望企业的 CIO、CEO 和 CSO 都能够从本书中获得他们需要的一些宝贵理念和实践指引。本书中的一些理念、方法和实践指南，笔者都在企业信息安全建设中与一些 CIO、CEO 经过讨论并达成共识。

刚走出校门的大学生、研究生，以及正在通过各种渠道（包括培训、实习等）来试图进入信息安全领域的学子或者工作者，相信可以从本书中获得系统的知识和工作的指南。本书中所阐述的知识、操作所用的案例都是在实际工作中精挑细选的，相信读者可以从中提前感受和体会到作为一名信息安全从业人员所需要具备的基本知识结构以及实际的技能。

# 作者简介

本书的笔者有多年500强企业从事信息安全技术研发和管理的实践经验，擅长于Linux安全的管理和研发，具有深厚的Linux安全基础理论知识和丰富的项目经验。本书由李洋主持编写，参与编写的作者还有柴泽楠、靳文佳、张晓明、江扬旺、康宇、宋继阳、吴廷勇、张恒、孙定隆、陈义勇、石依山、姚笃君、李刚、王博、李淞洋、吕远、孙悦、张雷、史思冰、田源、关威等。

由于作者水平有限，书中难免存在疏漏与不当之处，敬请专家和广大读者给予批评指正。欢迎大家通过我的51CTO官方blog（http://patterson.blog.51cto.com/）以及新浪微博（http://weibo.com/u/2358007797）与我互动交流。

# 内容编排

在内容的编排上，本书精挑细选，摒弃了企业应用小众化的FTP、Samba、NFS等，浓墨重彩在500强企业关注的信息安全框架、数据安全、通信安全、移动办公安全等专题上面。本书各章的内容安排如下。

**第1篇（安全运维理论及背景准备）——“知己知彼，百战不殆”**

第1章是本书的入门知识。对于一个500强企业的信息安全管理者来说，必须要对国际、国内的信息安全现状，尤其是企业信息安全面临的威胁，以及所面对的对手——黑客，都要有一个非常清醒的认识，本章即对这些内容进行入门性的介绍。

第2章分门别类地详细介绍如何在各个层次来使用信息安全的相关技术来应对安全威胁。

**第2篇（企业Linux安全运维规划及选型）——“凡事预则立，不预则废”**

第3章介绍企业的信息安全工作的思路，以及具体的Linux安全的实施内容。

第4章介绍企业在选择时要考虑到其发行套件、内核版本以及服务器选型等诸多问题。并且介绍如何解决为了节省人力、物力，而大规模地高效部署Linux的问题。

**第3篇（企业Linux安全运维实战）——“战略上藐视敌人，战术上重视敌人”**

第5章从文件系统、进程、用户管理、日志安全四个方面出发，介绍企业对于这四个方面的安全防护技术、工具和策略。

第6章针对目录/文件的访问控制方式，介绍更为灵活的通过访问控制列表（ACL）以及SELinux的企业Linux安全加固机制。

第7章详细介绍企业如何通过紧密布控，全面地保证开源企业Web服务器运维安全。

第8章从分析基础网络服务面临的安全风险出发，给出如何对这些服务进行安全防护的方法和技术。

第9章介绍如何使用数据防泄露、加密技术和工具保障企业数据安全。

第10章介绍VPN技术的基本原理和分类，并深入分析和探讨如何使用开源的VPN技术来保障企业的移动数据通信安全。

第11章详细介绍企业如何应用相关的工具来进行企业Linux服务器远程安全管理。

第12章针对Linux网络，介绍企业网络流量管理的主要技术、理念和工具。

第 13 章详细介绍企业级防火墙的部署及应用。

第 14 章详细介绍企业立体式入侵检测及防御体系、技术及工具的应用。

**第 4 篇（企业 Linux 安全监控）——“师夷长技以制夷”**

第 15 章介绍企业 Linux 系统及性能监控的常用工具和方法，并通过大量的实例来揭示企业是如何进行此项安全运维工作的。

第 16 章介绍如何使用优秀的开源网络监控工具辅助网络管理人员和信息安全工作人员进行网络监控和管理。

第 17 章介绍如何在 Linux 系统下通过端口和漏洞扫描来发现企业网络漏洞，从而为安全工作者提供相应的素材，以针对具体信息采取相应的措施来对该漏洞进行修补等。

**第 5 篇（企业 Linux 安全运维命令、工具）——“工欲善其事，必先利其器”**

第 18 章对企业级 Linux 内核配置、编译、安装、系统恢复等细节性的问题进行详细介绍。

第 19 章对一些比较常用的优秀开源工具进行介绍。

附录 A 挑选了百余种 Linux 中最为常见和重要的命令，给出的这些命令的功能说明和参数，以及语法使用都是严格地根据 Linux 中的 man 手册参考得来的，希望为用户提供日常使用和学习的参考之用。

附录 B 为读者提供了一些在企业信息安全管理中可能用到的非常有用处的网络工具。

## 致谢

作者首先由衷地感谢清华大学出版社的资深编辑栾大成为本书出版所作的大量耐心、细致的工作，他对本书的系统性、严谨性和科学性方面提了大量宝贵的意见；还要感谢父母和妻儿对我的支持，尤其是我可爱的女儿，他们给我的生活带来了太多的快乐和色彩，没有他们，这本书也不可能成功问世。

# 回复信息安全爱好者的一封信

## 来信

李博士您好，期望百忙之中能浪费您一些时间，关于追求技术的矛盾：

很早就关注了您的blog，您发表的很多文章都值得借鉴，令我钦佩。正在研读您的《网络协议本质论》，巩固知识体系！开门见山，首先介绍下我自己，我是一个执着于追求技术的人，两年前，因为信息安全专业而去了一所大专院校学习。现在，两年过去了，也学到了一些基础知识，不过还没形成体系，而且偏向网络这一块，下面把这两年我所自学的东西列出来：

1．网络方向：网络基础、TCP/IP（仅原理，代码实现未研究），思科的CCNA、部分NP+模拟器实践，考取了软考的网络工程师中级职称证书，考这个证书的主要目的是为了把网络的整个知识体系打下一个框架性的认识，力争融会贯通。

2．操作系统方向：熟悉Linux、RHCE、各类服务搭建、Linux下ShellScript的编写（服务器运维）、操作系统原理、Server 2003、数据库等。

3．Web安全方向：ASP/PHP/JSP及一些脚本编写、网站攻防、渗透测试等（这一块还没开始）。

4．编程方向：C/C++/汇编等（停留在皮毛阶段，主要精力花在了网络技术和服务器（Linux）应用技术这一块）。

学习定位：

1．主攻Linux，掌握常用命令的用法，熟悉Shell脚本的编写，能高效搭建各种常用服务及安全。（60%精力）

2．掌握汇编语言/C/C++的基础，有时间+数据结构。（20%精力，上半年）

3． 完成任务2后开始学习Web安全、渗透等攻防技术，部分可归纳在Linux学习范围内。（30%精力、下半年）

4．熟悉网络设备的配置及技术原理，至少有个感性的认识，如多区域OSPF、VPN、MPLS和防火墙。（10%精力）

大概就是这些，路几乎都是自己摸索出来的，与任晓辉的那张黑客分类进阶图的网络攻防方向差不多，这条路似乎有些偏离安全技术的核心方向，如逆向、反汇编调试、脱克、破解、外挂、反病毒、木马、内核编写等底层的东西，但真正要把这些东西学通，又需要相当长的时间去掌握，而我要面对现实问题，半年后差不多就要去相关的工作岗位应聘，做一些偏向网络应用方面的技术，必须先站稳脚跟，找到谋生的手段，出校门后会先做一些Linux服务器的运维相关工作来作为过渡职位，可能偏向网站安全，先以Linux为主攻方向，然后利用空闲时间学习安全技术，包括Linux内核，并把安全这个知识面打好，比如学习 CISP的课程体系，虽然广而不精，重理论、概念性知识，缺少实践，但我觉得形成一个比较广的安全知识体系并深入研究技术会更好些，然后转而做一些信息安全的相关工作（应该是涉及面比较广的，而不是纯做某一项技术的大牛，比

如内核、逆向、软件安全、反病毒软件编写等），也许技术涉及的不深，但这样的方向也是出于我自身的各种因素及发展而理性判断而决定的，黑客这些核心技术，我想作为一种业余爱好去研究（也许能在工作中作为辅助甚至转为优势）。事实如此，不过心有不甘，我也想做到技术的巅峰，以致经常产生矛盾心理，不知道前辈是如何衡量这些现实因素的？

我渴望黑客、安全技术的最高境界，只因起步较晚，能力有限，马上要面临现实生存问题，所以必须想个办法，两年来，我主要精力花在网络这一块（虽然本质上都是相通的，但也需有所侧重嘛），所以我的方向是先以网络安全为主，具体点，应该是Linux系统管理员，一个服务器架设与确保服务安全的管理员，然后倾向Web安全，以这个作为成为“大牛”的一个跳板，一个暂时的安身之地，算一个比较折中的方向吧，我觉得做这一块，既要懂操作系统、网络、数据库、C、汇编，还要懂计算机组成原理、脚本语言、算法、逻辑思维等，才能把Linux学得比较好，我想，花几年时间，等Linux做到一定程度，编程基础也相应上来了，那么就可以深入它的内核，读它的内核源代码，深入底层的学习，不代表未来就不往黑客的核心技术发展，包括软件安全，但我必须先花几年时间把“网络安全”这一块做好，做得比较“精通”（业余时间学软件安全技术）。

我就把应用性技术比作“外功”，而把底层技术，比如汇编，比作“内功”，只剩半年多时间，我需要先把“外功”练好，毕业后找个容身之地，然后利用运维工作较多的空闲时间，去修炼“内功”。

近期目标：力争做一个优秀的Linux系统管理员，服务器安全、架设这一块，先以Web安全为主。

但我会先花时间掌握编写高效的运维脚本能力，即Linux下的ShellScripts，进一步了解Linux的工作原理，以致节省工作时间，然后利用这些空闲时间去学习黑客所需要的技术知识，比如C、汇编、数据结构与算法等重要的基础知识，Web安全要做好，又要懂数据库，比如Oracle。

具体岗位：比如维护百度、淘宝等门户网站的安全。

期望李前辈能在百忙之中花一些时间来帮我理清下思路与矛盾，从内心感激您，谢谢！

一名追求提高技术实力而获得成就感和满足、不断奋斗中的所谓的“信息安全学子”的来信。

# 我的回复

这位朋友，你好！

看了你的信，我非常高兴，你能够在信息安全的这条道路上默默探索，慢慢前进，非常不容易，值得肯定！

下面我帮你理清一下思路。其实你的情况非常典型，很多号称在信息安全领域搞了多年的人，也没你想的这么多，这么透彻，我非常高兴，中国的信息安全人才还是不缺乏的。建议你从如下几个方面着手学习和发展。

（1）打牢基础，注重积累

信息安全方面涉及的领域非常广，本质上是一个交叉性学科，涵盖的内容包括操作系统、计算机网络、数据通信、数据库、密码学、程序语言与设计、编译原理、计算机体系结构等，而且对每一门的要求都不低，至少要理解原理，并能够举一反三。所谓安全，主要的就是敏锐的洞察力和判断力，使用这些基础知识去对系统、软件、网络、应用等进行安全与否的判断以及提出解决办法，是一个非常综合性的工作。因此，基础打得牢不牢，知识面广不广，直接决定了你以后的发展。可以多买一些基础性的结合应用的书籍来看，这样既不会枯燥乏味，也能迅速提高自己，我写的一些书籍大多都是两者结合，你可以参考使用。

（2）勤于实践，多多动手

信息安全同时又是一门实践性和操作性非常强的学科，从你所用的一些工具软件、反汇编技术等，都需要实践。当然，这个实践不能盲目地从网上下载一些工具，拿来就用。需要理解其原理、特性和用途，才能有的放矢，也能迅速提高。这就需要以自己牢固的基础知识为后盾，切忌盲目、好高骛远，一心只想使用工具，而忽略了自己在各方面的沉淀和积累。现在的工具非常多，但是切忌不要进行恶意的攻击行为，可以采用虚拟机等在虚拟的环境中进行，这是一个前提，也是一个信息安全工作人员的职业操守。

（3）善于总结，醍醐灌顶

工作这么多年，从数十个国家几千万项目的信息安全研发工作到近年来大型企业的信息安全管理工作，我最大的收获就是要善于总结，阶段性地总结。各类应用不断出现，其漏洞和风险越来越多，所暴露的安全问题也越来越多，是“头痛医头，脚痛医脚”，还是总结得出一些好的方法，对一类问题的解决办法或者解决方案效率高呢？显然是后者。但是做起来不是那么容易的，需要有前面基础学习和实践经验甚至教训的总结。现在中国信息安全的格局比较混乱，很多所谓的信息安全工作人员只注重一朝一夕的工具的使用、一次两次的所谓“攻击实战”，却不注重方法，不注重总结，这是非常低效的。可以看到，国外的信息安全从业人员，都有非常深厚的基础功底和实践经验，这值得我们好好借鉴。这样，经过一段时间，你就会有醍醐灌顶、豁然开朗的感觉，你就慢慢进步了，慢慢地朝着正规的、良性的方向发展。

因此，根据上面几个原则，相信不难规划你的信息安全之路，祝你学业有成！

李 洋

# 推 荐 序 一

“结识李先生已经 6 年有余了。从一名技术媒体人的角度来看，本书既不是空洞的纸上谈兵，也不是无来由的蠢蠢冒进，而是李先生在多年的工作和研究中所积累的理论和经验。

开源是所有的 IT 爱好者所热衷的话题，但是如何将开源合理、安全地利用到企业信息化建设，如何高质量地针对开源进行维护，如何利用开源有效地抵御和防范日渐增强的企业信息化威胁，诸如此类的问题是企业级信息化管理者需要不断攻克的一个课题。而本书正是为这些企业级信息化管理者带来了一套切实可行的方式方法。在此向各位读者朋友推荐这本书籍，也祝愿李先生的作品能够让更多的读者从中得到收获。

**——51CTO 网站副总编赵磊先生**

# 推 荐 序 二

安全问题，是这两年不断引爆人们神经的话题，从 QQ 诈骗，到某网站礼品卡被盗用，再到国内知名 IT 论坛用户名密码泄露，人们的信息安全和财产安全面临着一次又一次的危机，而发生这些事情后，我们第一个反应到的词是什么？是安全！

当我打开这本书的时候，看到的不是“一雨池塘水面平，淡磨明镜照檐楹”，而是“醉卧沙场君莫笑，古来征战几十还”，在系统一次次被攻陷，多少人陷入“但使龙城飞将在，不教胡马度阴山”的场景。没有刀光剑影的江湖，就不叫江湖。同样没有你攻我防的网络，也不是网络。网络的本意就是将大家的信息孤岛互相联系起来，融入一网，而称为互联网。而在这个圈子里面，总有人不甘寂寞，喜欢窥探其他人的“岛屿”。于是大家自愿自发地组建了生态圈自卫队，而这本书，我认为是自卫队的必读教材。

现在多数的书籍要么把重点都放在了概念阐述或者是学术理论上，忽略了实际操作的疑惑和应用难题的解决；要么将运维说得概念大到天，忽略了运维的真正意义。而该书打破了这种常规思路，给读者耳目一新的感觉。这本书是一本着眼于现在企业安全管理难题的实用型工具书。其理念对于企业 IT 管理人员具有指导意义，在使用任何工具前，翻阅一下这本书，它会告诉你在新系统上线前，可能会遇到哪些问题；也会告诉你在上线后，可能会受到哪些攻击。这本书以 Linux 为平台，详细讲述了安全运维理念中最为核心的方法。应用普及到安全边界、身份安全与访问管理、数据保密以及安全监控和管理等，几乎无所不容。

李洋已经在安全领域从事 10 余年，一直致力于计算机网络信息安全的研发工作，现任一家国际知名投资银行的资深信息安全顾问和架构师，同时担任 ICC、IEEE Communications Letters、Globecom、Elsevier Computer Communications、Elsevier Computers Security 等多种 SCI 检索期刊和国际著名会议的审稿人和 TPC，并在国家 863、国家自然科学基金、国家 242 信息安全计划项目等项目中作为主要成员，为项目组做出了显著贡献。他对于计算机安全技术和解决方案有着高度的热情和执着，这种热情和执着作为他生命中的特质伴随其职业生涯。也正是这种特质，使他

获得了大量的学术认可和项目成果，他在安全领域的大量著作就是他这种特质的一个缩写。李洋在网络安全领域拥有丰富的阅历，在写作方面也有着出众的引导读者的能力，经过他性格特质的加工，使得该书必然是一部海纳百川的书。如果您正在为公司的安全问题担忧，不妨考虑一下这本书，该书给读者带来的将不只是知识的累计和经验的提升，还希望读者在遇到问题的时候，通过这本书可以"众里寻他千百度，蓦然回首，答案只在桌上手边处"。

**——汇美国际下属科技公司 CTO 战剑新先生**

# 推 荐 序 三

Network security has been becoming the cornerstone of modern information system. With the widespread deployment of Cloud technology facilities and the extensive application of electronic settlement and payment in the field of finance service, this trend is accelerating, thus from this aspect, security has been at the heart of corporate information system.

Doctor Yang Li has been a senior expert with extensive experience in network security. This book is his another masterpiece in security field which is almost rated as the perfect combination of security theory and practice, to some extent, it is a bible book of security.

WORTHWHILE READING!!!

——Andrew Liu, the CTO of Mobile Payment Inc. @Silicon Valley

# 目 录

## 第一篇 安全运维理论及背景准备

## 第二篇 企业Linux安全运维规划及选型

# 第三篇 企业 Linux 安全运维实战

# 第四篇 企业 Linux 安全监控

# 第五篇 企业 Linux 安全运维命令、工具

# 第一篇
# 安全运维理论及背景准备

## 知己知彼，百战不殆

出自兵家:《孙子·谋攻篇》中说:“知己知彼，百战不殆；不知彼而知己，一胜一负；不知彼，不知己，每战必殆。” 意思是说，在军事纷争中，既了解敌人，又了解自己，百战都不会有危险；不了解敌人而只了解自己，胜败的可能性各半；既不了解敌人，又不了解自己，那么每战都会有危险。

安全运维工作需要有知己知彼的睿智，才能够在安全与威胁之争中取得先机，也能使企业在安全运维工作中立于不败之地。

# 第 1 章

# 知彼：企业信息安全现状剖析

## 本章导读

所谓“知己知彼，百战不殆”，对于网络安全工作者和 Linux 安全管理员来说尤其如此。在学会并使用相关的 Linux 安全技术之前，必须要对网络安全现状有一个全面、整体、宏观和清醒的认识。具体来说，需要明确以下几个方面的知识和背景。

- 国内和国际网络安全问题及其现状，包括一些相关的安全组织和标准。
- 黑客入侵问题的严重性。
- 黑客入侵所采用的常用攻击手段分类和基本原理。

对于一个 500 强企业的信息安全管理者来说，必须要对国际、国内的信息安全现状，尤其是企业信息安全面临的威胁，以及所面对的对手——黑客，要有一个非常清醒的认识，这是作好大型企业信息安全工作的第一步。

## 1.1 信息安全问题概览

目前越来越多的网络系统面临着攻击和入侵的威胁。CERT（Computer Emergency Response Team）权威数据表明，从1988年起，CERT报告的安全事件每年以指数级增长。

中国国家互联网安全中心2011年8月9日在大连举行的“计算机网络安全年会”上表示，2010年以来网络安全事件的跨境化特点日益突出，中国遭受的各类网络安全事件中有近半数来自境外。

2010年，国家互联网应急中心监测发现共近48万个木马控制端IP，其中有22.1万个位于境外，前两位分别是美国（占14.7%）、印度（占8.0%）；共有13782个僵尸网络控制端IP，有6531个位于境外，前三位分别是美国（占21.7%）、印度（占7.2%）和土耳其（占5.7%）。另据工业和信息化部互联网网络安全信息通报成员单位报送的数据，2010年在中国实施网页挂马、网络钓鱼等不法行为所利用的恶意域名半数以上在境外注册。

国家互联网应急中心去年协调境外网络安全组织和域名机构，处理了多起针对境内的恶意扫描、网络钓鱼等网络安全事件，得到美国、韩国、澳大利亚等国应急组织和“国际反网络钓鱼联盟”等组织的配合。总体上看，跨境网络安全事件呈现快速增长趋势，国际网络安全合作需进一步加强。同时，去年中国共有近3.5万家网站被黑客篡改，其中被篡改的政府网站达4635个，比2009年上升67.6%，政府网站安全防护较为薄弱。此外，金融行业网站频频遭遇“网络钓鱼”，成为不法分子骗取钱财和窃取隐私的重点目标。2010年，国家互联网应急中心共接收网络钓鱼事件举报1597件，较2009年增长33.1%。被仿冒的网站按事件次数排在前十位的有9家是金融或经济机构。其中，包括美国电子商务网站、中国香港汇丰银行、中国工商银行（601398，股吧）。今年以来，国家互联网应急中心加大了对仿冒境内金融机构网站的监测力度，并重点处置涉及中国农业银行（601288，股吧）、中国银行（601988，股吧）等金融机构的仿冒事件。7月，国家互联网应急中心自主监测发现的仿冒境内银行的网站域名为278个，多数为境外注册的域名。

在Internet初期，少数黑客仅仅出于好奇心或炫耀自己高深技术的目的进入未授权系统，而现在大量黑客因利益驱使盗用资源、窃取机密、破坏网络。同时由于网络的普及和黑客工具软件的流行，使得攻击网络所需的技术门槛下降，因而破坏性更大。图1-1显示了近20年间网络攻击复杂性和所需知识的趋势。

从图1-1可以看出，在20世纪80年代，入侵者一般是信息安全领域的专家，拥有深厚的专业知识和独特的入侵攻击手段，很少依靠专用入侵工具。他们一般手工编写入侵代码，对目标系统进行攻击。但是现今由于大量的入侵工具能够从Internet上轻松获取，攻击者可以使用这些工具来攻击数以万计的系统。与此同时，攻击类型日益复杂，攻击手段趋向于多样化，入侵和破坏在瞬间完成，同时采取隐藏行踪的手段来逃避检测系统的跟踪。例如，20世纪80年代和90年代初期，DoS事件（拒绝服务攻击）较少报道，并未引起重视，而现在对于电子商务、在线证券交易等网络贸易，DoS攻击经常造成系统停止运作。另外，在攻击频率高速增长的同时，攻击者具备的专业知识总体在下降，攻击手段却日益复杂，各网络系统都面临着严峻考验。据报道，世界上平均每20秒就发生一起黑客入侵事件，在美国每年因此造成的经济损失高达100多亿美元，涉及政府机构、军事国防、科研院校、金融商业等各部门。计算机网络犯罪已严重干扰了人们的正常生活，造成巨大的经济损失，直接或间接地威胁着国家安全。

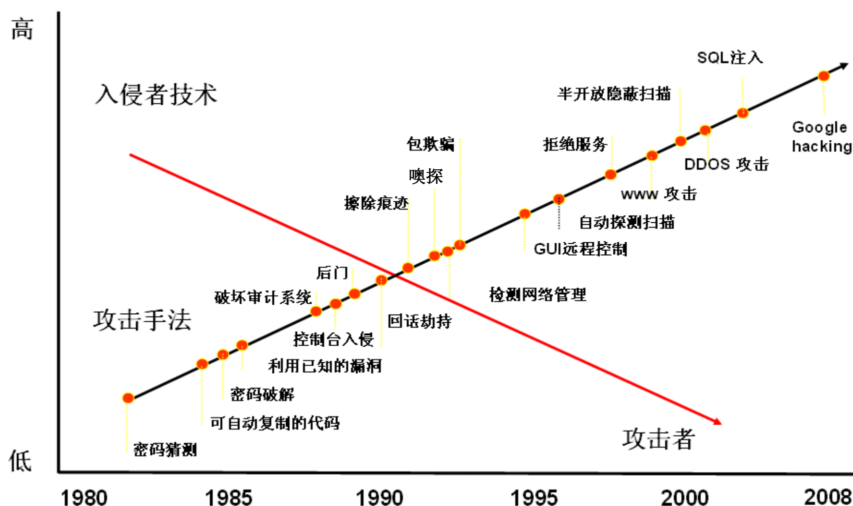



图 1-1　入侵攻击的复杂性与入侵者所需知识程度的关系

根据最新的国家计算机网络应急技术处理协调中心报告，目前网络攻击的动机逐渐从技术炫耀型转向利益驱动型，网络攻击的组织性、趋利性、专业性和定向性继续加强，从而导致为获得经济利益的恶意代码和在线身份窃取成为网络攻击的主流，瞄准特定用户群体的定向化信息窃取和勒索成为网络攻击的新趋势。根据已有资料分析，目前网络攻击从整体上呈现以下三个新的特点。

- 攻击组织严密化：黑客逐步形成较为严密的组织，致使网络攻击的效率有明显提高。
- 攻击行为趋利化：针对商业竞争对手的攻击和用于窃取用户账号、密码等敏感数据的网络攻击逐渐增多。随着网络行为同社会行为联系的进一步密切，网络攻击的最终目的越来越多地落在获取具体的经济利益上。
- 攻击目标直接化：网络黑客针对攻击目标的特点，设计特定的攻击代码，绕过网络防御体系入侵有价值的目标主机，或者通过僵尸网络对目标发起直接的大规模网络攻击，使得针对特定目标的网络攻击具有更大的威胁和破坏性。

目前，信息安全问题主要体现在以下几个方面。

### 1.1.1　黑客入侵

计算机信息网络上的黑客攻击事件越演越烈，已经成为具有一定经济条件和技术专长的形形色色攻击者活动的舞台。他们具有计算机系统和网络脆弱性的知识，能使用各种计算机工具。境内外黑客攻击破坏网络的问题十分严重，他们通常采用非法侵入重要信息系统，窃听、获取、攻击侵入网的有关敏感性重要信息，修改和破坏信息网络的正常使用状态，造成数据丢失或系统瘫痪，给国家造成重大政治影响和经济损失。在计算机网络技术发展和互联网应用日益普及的今天，黑客技术也日益高明。目前，黑客能够运用的技术手段和攻击软件已经高达上千种，并呈日渐增长之势。并且，黑客也在开发或者利用已有的工具来寻找计算机系统或者是网络的缺陷和漏洞，并针对其进行有目的的攻击。并且，从图 1-1 也可以清晰地看到，黑客工具可以通过互联网轻而易举地获得，也使得黑客的入侵更加方便和快捷，网络安全形势日趋严重。黑客问题的出现，并非黑客能够制造入侵的机会，从没有路的地方走出一条路，而是因为他们善于发现漏洞。即信息网络本身的不完善性和缺陷，成为被攻击的目标或被利用为攻击的途径，其信息网络的脆弱性引

发了信息社会的脆弱性和安全问题，并构成了自然或人为破坏的威胁。

### 1.1.2 病毒发展趋势

随着计算机网络技术的发展。计算机病毒技术也在快速地发展和变化之中，而且在一定程度上走在了计算机网络安全技术的前面。专家指出，从木马病毒的编写、传播到出售，整个病毒产业链已经完全互联网化。对数量继续暴增的计算机病毒来说，防护永远只能是一种被动防护，而计算机感染上病毒后，轻则使系统工作效率下降，重则造成系统死机或毁坏，使部分文件或全部数据丢失，甚至造成计算机主板等部件的损坏，导致硬件系统完全瘫痪。据公安部调查结果显示，计算机病毒仍然呈现出异常活跃的态势，互联网站被大量“挂马”，成为病毒木马传播的主要方式。同时，目前计算机病毒、木马等绕过安全产品的扫描、查杀甚至破坏安全产品的能力也增强了。可见，当前计算机系统遭受病毒感染的情况相当严重。

### 1.1.3 内部威胁

由于网络和计算机系统固有的特性，它在提高数据与设备共享性的同时，也带来了信息、数据被非法窃取、复制、使用等弊端，对涉及国家机密及企业内部敏感数据的安全管理形成极大的挑战。为防止数据外泄，企业往往不惜花巨资购进防火墙、入侵检测、防病毒、漏洞扫描等网络安全产品，以为这样就可以高枕无忧了。其实，这种想法是错误而且极其危险的。fbi 和 csi 对 484 家公司进行了网络安全专项调查，调查结果显示：超过 85%的安全威胁来自公司内部。在损失金额上，由于内部人员泄密导致了 6056.5 万美元的损失，是黑客造成损失的 16 倍，是病毒造成损失的 12 倍。这组数据充分说明了内部人员泄密的严重危害，同时也提醒国内组织应加强网络内部安全建设。上网单位由于对内部威胁认识不足，所采取的安全防范措施不当，导致内部网络安全事故逐年上升。不论是有意的还是偶然的，内部威胁将继续是一个最大的安全威胁。如果网络的安全策略是未知的或不能执行的，用户诸如冲浪不安全的网站，点击电子邮件中的恶意链接，或者不对敏感数据加密等行为都将继续不知不觉地扮演着安全炸弹的角色。而随着人员的移动性越来越强。利用未加密的移动设备使用网络也大大增加“暴露”的风险，给犯罪分子留下可乘之机。另外，一机两用甚至多用情况普遍，计算机在内外网之间频繁切换使用，许多用户将在 Internet 网上使用过的计算机在未经许可的情况下擅自接入内部局域网络使用，造成病毒的传入和信息的泄密。公安部调查结果显示，攻击或病毒传播源来自内部人员的比例同比增加了 21%，涉及外部人员的同比减少了 18%，说明联网单位绝大部分都是出于防御外部网络攻击的考虑，导致来自内部的威胁同时呈上升态势。通常内部威胁会造成致命后果。

### 1.1.4 自然灾害

计算机信息系统仅仅是一个智能的机器，易受自然灾害及环境（温度、湿度、振动、冲击、污染）的影响。目前，我们的不少计算机房并没有防震、防火、防水、避雷、防电磁泄漏或干扰等措施，接地系统也疏于周到考虑，抵御自然灾害和意外事故的能力较差。日常工作中因断电而设备损坏、数据丢失的现象时有发生。由于噪音和电磁辐射，导致网络信噪比下降，误码率增加，信息的安全性、完整性和可用性受到威胁。

## 1.2 各经济大国安全问题概要

### 1．美国信息安全现状

据美国联邦调查局统计，美国每年因网络安全造成的损失高达 75 亿美元。据美国金融时报报道，世界上平均每 20 分钟就发生一次入侵国际互联网络的计算机安全事件，三分之一的防火墙被突破。美国联邦调查局计算机犯罪组负责人吉姆•塞特尔称：给我精选 10 名“黑客”，组成小组，90 天内我将使美国趴下。一位计算机专家毫不夸张地说：如果给我一台普通计算机、一条电话线和一个调制解调器，就可以令某个地区的网络运行失常。

目前在信息安全技术处于领先的国家主要有美国、法国、以色列、英国、丹麦、瑞士等，一方面这些国家在技术上特别是在芯片技术上有着一定的历史沉积，另一方面在这些国家的信息安全技术的应用上例如电子政务、企业信息化等起步较早，应用比较广泛。其领先优势主要集中在防火墙、入侵监测、漏洞扫描、防杀毒、身份认证等传统的安全产品上。而在注重防内兼顾防外的信息安全综合强审计上，国内的意识理念早于国外，产品开发早于国外，目前在技术上有一定的领先优势。

美国“9.11”以后，国际网络安全学术研究受到国际大气候的影响，围绕“反恐”的主题展开了许多的工作，但工作的重心是以防止外部黑客攻击为主。实际上“恐怖分子”大多是在取得合法的身份以后再实施恶意攻击和破坏的。审计监控体系正是以取得权限进入网络的人的操作行为都是不可信任的为前提假设，对所有内部人的操作行为进行记录、挖掘、分析，从而获得有价值信息的一套安全管理体系，目前在国际上具有领先地位。

构建可信的网络，建设有效的信息安全保障体系，实施切实可行的信息安全保障措施已经成为世界各国信息化发展的主要需求。信息化发展较高的发达国家，非常重视国家信息安全的管理工作，如美、俄、德、日等国家都已经或正在制订自己的信息安全发展战略和计划，确保信息安全沿着正确的方向发展，他们在信息安全领域进行着积极有益的探索。

美国的信息化程度全球最高，是信息超级大国，在信息技术的主导权和网络上的话语权等方面占据先天优势，他们在政府信息系统的信息安全体系建设以及政策支持方面也走在全球的前列。美国信息安全管理的最高权力机构是美国国土安全局，分担信息安全管理和执行的机构有国家安全局、联邦调查局、国防部、商务部等，主要根据相应的方针和政策结合自己部门的情况实施信息安全保障工作。

美国国防部几乎影响了全世界的信息安全概念、观念和理念。1967 年，DOD 开始研究计算机安全问题，1977 年提出加强联邦政府和国防系统计算机安全的倡议，1983 年提出可信计算机系统评估准则（TCSEC），1987 年对新发布的《计算机安全法》的执行情况进行部门级评估，1997 年发布《国防部 IT 安全认证认可规程》（DITSCAP），该规程在 2000 年由国家安全委员会发布为《国家信息保障认证和认可规程》（NIACAP）。目前，美国已制定的有关信息安全的法律有：信息自由法、个人隐私法、计算机安全法、电信法、联邦信息安全管理法案等，形成了较完备的信息安全保障体系。

2002 年美国颁布的《联邦信息安全管理法案》（FISMA），试图通过采取适当的安全控制措施来保证联邦机构的信息系统安全性，是当前美国信息安全领域的一个重要发展计划。FISMA 的实施分为三个阶段，分别为开发标准和指南（2003—2008）、形成安全能力（2007—2010）和运用

自动化工具（2008—2009）。为了有效地实现FISMA目标，FISMA指定美国国家标准与技术研究所（NIST）开发和发布相关的标准、指导方针以及其他出版物，帮助联邦机构实现联邦信息安全管理法案，以保护其信息和信息系统。作为FISMA第一阶段的成果，NIST提供了一套有效的信息安全保障框架和机制，提供了保护信息和信息系统的有效方法和手段，促成了一套全面权威的信息安全标准体系，其中包括《IT系统安全自评估指南》（SP 800-26）、《IT系统风险管理指南》（SP 800-30）、《联邦IT系统安全认证和认可指南》（SP 800-37）、《联邦信息和信息系统的安全分类标准》（FIPS 199）、《联邦IT系统最低安全控制推荐》（SP 800-53）、《将各种信息和信息系统映射到安全类别的指南》（SP 800-60）、《IT产品国家配置清单项目——配置清单用户和开发者指南（草案）》（SP 800-70 Revision 1）（Draft）等多个文档，以风险管理思想为基础加强联邦政府的信息安全，对设计和实现有效的信息安全项目具有十分重要的意义。所有美国联邦政府机构以及承包商或其他代表联邦机构的组织所使用或管理的信息系统都被强制要求遵循NIST发表的SP800系列、联邦信息处理标准（FIPS）文件，以及其他联邦信息系统的相关法律条例。

FISMA第二阶段的工作目标是要形成安全能力，通过评估拿出符合性凭据，美国国家标准技术委员会发布了若干个不同的标准，包括150、150-17和7328等，主要目标就是约束相关的信息安全测评机构，需要具备这样的服务能力和服务准则来为联邦相关机构的信息系统进行测评，以确认这些系统是否符合信息安全管理法案的要求。

FISMA最后一个阶段的工作重点是落地，其工作目标就是，推动自动化工具的使用，从2009年向后的3～5年的时间里，尽可能地把一些规范和标准自动化、工具化，从而配合安全运维人员更好地工作。NIST推出了S-CAP协议（安全内容自动化协议），它是一种通过明确的标准化的模式使漏洞管理、安全监测和政策符合性能够与美国联邦信息安全管理法案一致的方法。主要由六个不同的技术标准（CVE、CCE、CPE、XCCDF、OVAL和CVSS）作为支撑，六个不同的标准分别从漏洞、配制、系统脆弱性的统一命名，包括脆弱性严重性的评分标准，安全检查的步骤，检查项目及检查报告等各个方面进行有效地设定，从而保证后期的工具、软件开发厂商能够有一个明确可落地的标准进行参考。在S-CAP的基础上由美联邦政府牵头针对一些联邦政府的桌面主机开展了一个FDCC的安全项目，专门对Windows系统主机的安全漏洞和安全配置进行检查的标准，并由安全厂商开发了对应的FDCC Scanner设备进行自动化的检查，而FDCC也为通过FDCC认证的Scanner颁发证书。

2007年，美国IT投资650亿美元，安全投资59亿美元，占9.2%；2009年美国将加大政府信息系统安全投入，IT支出达到709亿美元，其中的10.3%，即总额约73亿美元将用于提高IT安全性。

### 2. 俄罗斯信息安全现状

俄罗斯十分重视信息安全的作用，时时处处以信息安全危机来鞭策、督促各单位把信息安全工作做得更好。从法令、机构人员、资金、技术、管理等角度全方位给信息安全检查工作予以支持和保障。政府发布了许多有关信息安全的法律法规。例如1995年，俄罗斯宪法把信息安全纳入国家安全管理范围，颁布了《联邦信息、信息化和信息网络保护法》，强调国家在建立信息资源和信息网络化中的责任，1997年的《俄罗斯国家安全构想》明确提出保障国家安全应把保障经济安全放在第一位，而信息安全又是经济安全的重中之重。2002年，俄罗斯安全委员会通过《国家信息安全学说》，明确了联邦信息安全建设的目的、任务、原则和主要内容，对国家信息网络安全面临的问题及信息网络战武器装备现状、发展前景和防御方法等进行了详尽的论述，阐

明了俄罗斯在信息网络安全方面的立场、观点和基本方针，提出在该领域实现国家利益的手段和相关措施。第一次明确指出俄罗斯在信息领域的利益是什么、受到的威胁是什么以及为确保信息安全所要采取的措施等。

随着全球信息化步伐的加快，俄罗斯对国家信息安全的重视程度日益提高，信息网络安全已纳入国家安全战略。普京总统强调："信息资源和信息基础设施已经成为争夺世界领先地位的舞台，未来的政治和经济将取决于信息资源。因此，解决这方面的问题，对国家的前途、国家利益和国家安全至关重要。"

俄罗斯建立了完善的信息保护国家系统，通过执行俄罗斯联邦总统直管的国家技术委员会条例，保证信息保护领域的国家统一政策，同时兼顾国家、社会和个人利益的均衡。俄罗斯联邦政府从信息安全、经济安全、国防安全、生态安全和社会安全几个方面入手，将信息安全策略分为全权安全政策和选择性安全政策两类，提出了主客体分级访问的构想来控制存取访问，即：只有当主体的现时安全能力不低于客体临界标记时，信息方可"向上"传输。

俄罗斯联邦政府联络与情报局为俄罗斯联邦国家政权机关建立了因特网网段——RGIN（Russian Government Internet Network）。他们在保障信息安全方面还做了大量的工作。首先建成了高效安全的"阿特拉斯"数据传输，确保俄罗斯联邦各主体行政中心之间文件的网络传输，在最高国家机关安装了保障加密数据交换的技术设备，解决了该系统与国内其他通信网协同的技术课题。然后他们还确立了《计算机系统安全评估标准》、《产品安全评估软件》等一系列完善的系统安全评估指标。同时，建立了联邦经济信息保护中心，负责政府网络及其他的专门网络、网络信息配套保护、国家政权机关信息技术的保障等。

俄罗斯在发展信息安全技术上坚持自主创新、自成体系。强调数学模型与论证发挥自动控制理论的作用。注重芯片和操作系统的研发。俄罗斯的芯片设计技术独具风格，目前已达到世界领先水平。操作系统是俄罗斯大学和科研机构重点攻关的课题。如圣彼得堡技术大学已研制了自主安全内核的高安全等级操作系统，不受病毒和黑客的侵犯，是在安全的数学模型基础上进行开发的。在与国外产品兼容上只局限于外层的功能调用，内核是自己的。

此外，俄罗斯联邦政府在保障财政信贷和银行领域信息安全方面做了大量的工作。中央银行系统研究了保护信息处理技术设备的问题，积极推广使用有安全保护的信息技术和网络技术。在加密领域，密码学院从事着加密技术的研究工作，着力加强光纤通信加密和量子加密方面的研究。

### 3. 德国信息安全现状

德国是欧洲头号经济大国，也是仅次于美国、日本的世界第三经济强国。通过制定和实施信息化发展战略，德国信息化获得了较快的发展。德国在信息安全方面是欧洲的典范，其主要做法包括以下三方面。

一是有明确的责任部门。德国联邦经济和劳工部下属的联邦电信和邮政总局主要负责联邦电信基础设施的安全维护工作；内政部和其下属联邦信息安全署主要负责信息技术应用方面的安全问题，如互联网安全管理、防病毒入侵和应急处理计算机问题等。联邦安全署还负责对互联网进行内容监管，对需要跨部门协调的工作制定统一的方案。联邦内政部下属的联邦信息安全署设有计算机紧急反应小组，提供每天 24 小时的"应急服务"，解决互联网的安全问题，防止计算机病毒和网络攻击。联邦政府专门成立联邦信息安全办公室，负责处理信息安全方面的技术问题。

二是重视运用法律手段。德国联邦经济和劳工部下属的联邦电信和邮政总局在为德国联邦其

他部门提供基础电信服务的同时，还负责起草和制定《电信法》和《数字签名法》等法律，并协调联邦政府各部门有效使用数字签名来保障信息安全。联邦政府制定了具体的计划和措施加强互联网上的安全，包括颁布了《电子签名法》和《电子商务法》。

三是综合运用相关技术措施。德国联邦政府为加强信息安全采取了一系列的措施，包括重大基础设施的保护，增强社会各界的信息安全意识，通过设立安全门户网站为企业和个人提供相关信息和安全工具，增强互联网上的信息安全，开展信息安全认证，推广新的安全技术，与 IT 企业合作开展安全技术趋势研究，大力研发和使用密码技术、安全可靠的构件和生物识别技术等。

#### 4. 日本信息安全现状

日本自 2001 年 1 月《IT 基本法》颁布以来，在根据该法制订的每一年重点发展计划中，对电子政府以及电子政务相关的具体内容都做出了规定，针对计算机网络信息安全对策的体制，专门制订了一套相应的规则。以制订－引入－运用－评价－修正的模式，循环往复周期运行，针对风云变幻的计算机信息安全领域出现的新问题，进行及时有效的应对，可以说是一个值得称道的运作模式。

日本中央政府各个部委基于对政府信息系统的安全问题考虑，要求结合本单位、本部门的具体特点，考虑制定相应的规则、办法。

为了确保信息安全制度的有效实施，日本政府设立了相应的信息安全机构，明确了包括信息安全中心、政府各个部委在内的多个机构的职责。日本政府要求政府的各个部委应当依法加强对计算机信息安全的管理与控制，防止利用政府的计算机系统对其他的计算机系统进行恶意攻击事件的发生。为了进一步提高日本的计算机信息安全水平，要求政府不断加强与民间团体、企业研究机构之间的信息交换，建立官民紧密协作、联动的合作体制。

日本总理府办公厅信息安全中心负责制定《政府机关关于信息安全的统一基准》，要求各个部委应当结合本部门计算机信息安全具体规则实施的情况、特点，对来自外部网络的计算机技术性的威胁等情况，进行持续不间断的研究、评价，在此基础上制定相应的规则。

在计算机信息安全规则的制定中，从组织机构、基本方针的设定、风险的预测、对策基准的制定、信息的分类及管理、违反信息安全行为的应对措施几个方面提出了具体要求。

对在业务中存在违反信息安全行为的情形，提出了有必要建立依据上级的指示，直接命令其停止使用网络终端机器的体制。

#### 5. 我国信息安全现状

据了解，从 1997 年底至今，我国的政府部门、证券公司、银行等机构的计算机网络相继遭到多次攻击。公安机关受理各类信息网络违法犯罪案件逐年剧增，尤其以电子邮件、特洛伊木马、文件共享等为传播途径的混合型病毒愈演愈烈。由于我国大量的网络基础设施和网络应用依赖于外国的产品和技术，在电子政务、电子商务和各行业的计算机网络应用尚处于发展阶段，以上这些领域的大型计算机网络工程都由国内一些较大的系统集成商负责。有些集成商仍缺乏足够专业的安全支撑技术力量，同时一些负责网络安全的工程技术人员对许多潜在风险认识不足，缺乏必要的技术设施和相关处理经验，面对形势日益严峻的现状，很多时候都显得有些力不从心。也正是由于受技术条件的限制，很多人对网络安全的意识仅停留在如何防范病毒阶段，对网络安全缺乏整体意识。

我国已初步建成国家信息安全组织保障体系。国务院信息办专门成立了网络与信息安全领导小组，各省、市、自治州也设立了相应的管理机构。2003 年 9 月，中共中央办公厅、国务院办

公厅转发的《国家信息化领导小组关于加强信息安全保障工作的意见》（简称 27 号文），将信息安全提到了促进经济发展、维护社会稳定、保障国家安全、加强精神文明建设的高度，并提出了“积极防御，综合防范”的信息安全管理方针。2004 年 9 月 15 日，由公安部、国家保密局、国家密码管理局和国信办联合下发的《关于信息安全等级保护工作的实施意见》（66 号文件）明确了实施等级保护的基本做法。2007 年 6 月 22 日又由四部委联合下发《信息安全等级保护管理办法》（43 号文件），规范了信息安全等级保护的管理。2007 年我国基本完成了重要信息系统和基础信息网络的等保定级，等级保护工作即将进入建设整改阶段。

制定和引进了一批重要的信息安全管理标准，包括：《计算机信息系统安全保护等级划分准则》（GB 17895—1999）、《信息安全技术 信息系统安全管理要求》（GB/T 20269—2006）、《信息安全技术 信息系统安全工程管理要求》（GB/T 20281—2006）、《信息安全技术 信息系统安全等级保护基本要求》（GB/T 22239—2008）、《信息安全技术 信息系统安全等级保护定级指南》（GB/T 22240—2008）、《信息安全管理实施细则》（ISO17799:2005）和《信息安全管理体系要求》（ISO27001:2005）等。

制定了一系列必需的信息安全管理的法律法规。从 20 世纪 90 年代初起，为配合信息安全管理的需要，国家相关部门、行业和地方政府相继制定了《中华人民共和国计算机信息网络国际联网管理暂行规定》、《商用密码管理条例》、《互联网信息服务管理办法》、《计算机信息网络国际联网安全保护管理办法》、《电子签名法》等有关信息安全管理的法律法规文件。

开展了信息安全风险评估工作，并作为信息安全管理的核心工作之一，由国家信息中心组织先后对四个地区（北京、广州、深圳和上海），十几个行业的 50 多家单位进行了深入细致的调查与研究，最终形成了《信息安全风险评估调查报告》、《信息安全风险评估研究报告》和《关于加强信息安全风险评估工作的建议》。制定了《信息安全技术 信息安全风险评估规范》（GB/T 20984—2007）。

目前我国在信息安全方面存在的问题主要如下。

- 信息安全管理比较混乱，缺乏国家层面上权威、统一的整体组织和策略，缺乏专门的组织、规划、管理和实施协调的立法管理机构，执法主体不明确，多头管理，规则冲突，可操作性差，执行难度较大。实际管理、政策执行和监督力度不够。
- 具有我国特点的、动态的和涵盖组织机构、文件、控制措施、操作过程和程序及相关资源等要素的信息安全管理体系尚未建立起来。为了推动等级保护工作的持续发展和深化落实，我们急需提出清晰的等级保护工作和标准体系，准备好评估检查的能力。
- 具有我国特点的信息安全风险评估标准体系有待完善，信息安全的需求过于抽象，缺乏系统、全面的信息安全风险评估和评价体系，以及全面、完善的信息安全保障体系。
- 信息安全意识缺乏，普遍存在重产品、轻服务，重技术、轻管理的思想。
- 专项经费投入不足，管理人才缺乏，基础理论和关键技术研究、标准制定能力薄弱，严重依靠国外，在国际上缺乏话语权。
- 整体安全防范技术水平低下，尤其是核心技术方面（如：CPU 和操作系统），技术创新不够，信息安全管理产品水平和质量不高。
- 缺乏支撑信息安全管理标准落地的有效手段。

总的来说，我国信息系统的安全现状不容乐观，信息系统存在很多安全隐患。与西方发达国家相比，我国的信息安全起步较晚，政府部门和民众对网络认识不深，政府部门对信息系统安全的重视程度不够，安全意识淡薄，信息系统的网络与信息安全防护能力仍处于“初级阶段”，甚

至许多外网网站处于“不设防”状态。

## 1.3 企业面临的主要信息安全威胁

### 1.3.1 扫描

对位于网络中的计算机系统来说，一个端口就是一个潜在的通信通道，也就是一个入侵通道。对目标计算机进行端口扫描，能得到许多有用的信息，从而发现系统的安全漏洞。通过其可以使系统用户了解系统目前向外界提供了哪些服务，从而为系统用户管理网络提供了一种参考手段。

从技术原理上来说，端口扫描向目标主机的 TCP/UDP 服务端口发送探测数据包，并记录目标主机的响应。通过分析响应来判断服务端口是打开还是关闭，就可以得知端口提供的服务或信息，如图 1-2 所示。端口扫描也可以通过捕获本地主机或服务器的流入流出 IP 数据包来监视本地主机的运行情况，不仅能对接收到的数据进行分析，而且能够帮助用户发现目标主机的某些内在的弱点，而不会提供进入一个系统的详细步骤。

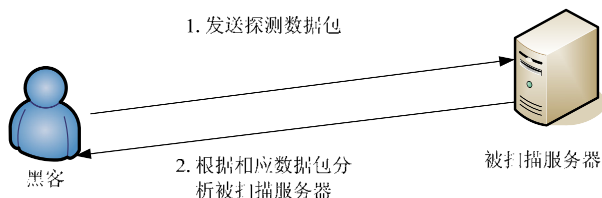



图 1-2　扫描过程示意

一般来说，端口扫描的目的通常是以下一项或者多项。

- 发现开放端口：发现目标系统上开放的 TCP 或 UDP 端口。
- 了解主机操作系统信息：端口扫描可以通过操作系统的“指纹”来推测被扫描操作系统或者应用程序的版本等信息。
- 了解软件或者服务版本：软件或服务版本可以通过“标志获取”或者应用程序的指纹来识别获得。
- 发现脆弱的软件版本：识别软件和服务的缺陷，从而有助于发起针对漏洞的攻击。

端口扫描主要有经典的扫描器（全连接）以及所谓的 SYN（半连接）扫描器。此外还有间接扫描和秘密扫描等。TCP 扫描方式是通过与被扫描主机建立标准的 TCP 连接，因此这种方式最准确，很少漏报、误报，但是容易被目标主机察觉、记录。SYN 方式是通过与目标主机建立半打开连接，这样就不容易被目标主机记录，但是扫描结果会出现漏报，在网络状况不好的情况下这种漏报是严重的。

### 1.3.2 特洛伊木马

特洛伊木马源于古希腊的特洛伊木马神话（见图 1-3），传说希腊人围攻特洛伊城，久久不能得手。后来想出了一个木马计，让士兵藏匿于巨大的木马中。大部队假装撤退而将木马摈弃于特洛伊城，让敌人将其作为战利品拖入城内。隐藏在木马内的士兵则乘夜晚敌人庆祝胜利、放松

警惕的时候从木马中爬出来，与城外的部队里应外合而攻下了特洛伊城。



图 1-3　特洛伊木马图示

在计算机网络安全领域，特洛伊木马通常是指一个包含在合法程序中的非法程序。该非法程序被用户在不知情的情况下执行。一般的木马都有客户端和服务器端两个执行程序，其中客户端是用于攻击者远程控制植入木马的机器，服务器端程序即是木马程序。攻击者要通过木马攻击系统，它所做的第一步是要把木马的服务器端程序植入到被攻击用户的电脑中。

目前木马入侵的主要途径还是先通过一定的方法把木马执行文件存入到被攻击者的电脑系统里，如邮件、下载等，然后通过一定的提示，故意误导被攻击者打开执行文件，比如故意谎称这是个木马执行文件用户朋友送给的贺卡，当用户打开这个文件后，确实有贺卡的画面出现，但这时木马可能已经悄悄地在被攻击者的后台运行了。

一般的木马执行文件非常小，大都是几 KB 到几十 KB，如果把木马捆绑到其他正常文件上，你很难发现的，所以，一些网站提供的软件下载往往捆绑了木马文件，在用户执行这些下载的文件时，也同时运行了木马。木马也可以通过 Script、ActiveX 及 ASP、CGI 交互脚本的方式植入，木马在被植入攻击主机后，一般会通过一定的方式把入侵主机的信息，如主机的 IP 地址、木马植入的端口等发送给攻击者，当攻击者获得这些信息后就能够与木马里应外合控制攻击主机。

在早期的木马里面，大多都是通过发送电子邮件的方式将入侵主机信息告诉攻击者的，有一些木马文件干脆把主机所有的密码用邮件的形式通知给攻击者，这样攻击者就不用直接连接攻击主机即可获得一些重要数据，如攻击 OICQ 密码的 GOP 木马即是如此。使用电子邮件的方式对攻击者来说并不是最好的一种选择，因为如果木马被发现，可以通过这个电子邮件的地址找出攻击者。现在还有一些木马采用的是通过发送 UDP 或者 ICMP 数据包的方式通知攻击者。

木马主要有以下几种类型，用户须多加注意。

- 破坏型。惟一的功能就是破坏并且删除文件，可以自动地删除电脑上的 DLL、INI、EXE 文件。
- 密码发送型。可以找到隐藏密码并把它们发送到指定的信箱。有人喜欢把自己的各种密码以文件的形式存放在计算机中，认为这样方便；还有人喜欢用 Windows 提供的密码记忆功能，这样就可以不必每次都输入密码了。许多黑客软件可以寻找到这些文件，把它们送到黑客手中。也有些黑客软件长期潜伏，记录操作者的键盘操作，从中寻找有用的密码。
- 远程访问型。最广泛的是特洛伊马，只要有人运行了服务端程序，如果客户知道了服务端的 IP 地址，就可以实现远程控制。这类远程控制程序可以实现监控被攻击者在系统中的一举一动，危害非常大，比如早些年一直流行的“冰河”木马软件。这类程序多数采用 UDP 协议，是因特网上广泛采用的通信协议之一。与 TCP 协议不同，它是一种非连接的传输协议，没有确认机制，可靠性不如 TCP，但其效率却比 TCP 高，因此其用于远程屏幕监视还是比较适合的。
- 键盘记录木马。这种特洛伊木马非常简单，通常只完成一件事情，就是记录被攻击者的键盘敲击并且在相应的日志文件里查找密码。这种特洛伊木马随着系统的启动而启动，有在线和离线记录这样的选项，分别记录被攻击用户在线和离线状态下敲击键盘

时的按键情况。从这些按键中，木马植入者可以很容易地得到被攻击者的密码等有用信息，甚至信用卡账号。当然，对于这种类型的木马，邮件发送功能也是必不可少的。

- DoS 攻击木马。随着 DoS 攻击越来越广泛的应用，被用作 DoS 攻击的木马也越来越流行起来。当攻击者成功入侵了一台机器，他通常会给这台机器种上 DoS 攻击木马，那么日后这台计算机就成为攻击者实行 DoS 攻击的最得力助手了。攻击者控制的肉鸡数量越多，那么其发动 DoS 攻击取得成功的机率就越大。所以，这种木马的危害不是体现在被感染计算机上，而是体现在攻击者可以利用其来攻击一台又一台计算机、给网络造成很大的伤害和带来损失上。还有一种类似 DoS 的木马叫做邮件炸弹木马，一旦机器被感染，木马就会随机生成各种各样主题的信件，对特定的邮箱不停地发送邮件，一直到对方瘫痪，不能接受邮件为止。
- 代理木马。黑客在入侵的同时掩盖自己的足迹，谨防别人发现自己的身份是非常重要的，因此，给被控制的“肉鸡”种上代理木马，让其变成攻击者发动攻击的跳板就是代理木马最重要的任务。通过代理木马，攻击者可以在匿名的情况下使用 Telnet、ICQ、IRC 等程序，从而隐蔽自己的踪迹。“肉鸡”是指被黑客专门用来描述 Internet 上那些防护性差、易于被攻破而且控制的计算机。
- FTP 木马。这种木马可能是最简单和古老的木马了，其惟一的功能就是打开 21 端口，等待用户连接。现在新 FTP 木马还加上了密码功能，这样，只有攻击者本人才知道正确的密码，从而进入对方的计算机。
- 程序杀手木马。上面介绍的木马功能虽然形形色色，不过到了对方机器上要发挥自己的作用，还要首先躲避防木马软件的查杀这一关才行。常见的防木马软件有 ZoneAlarm、Norton Anti-Virus 等。程序杀手木马的功能就是关闭对方机器上运行的这类防木马程序，让其他的木马更好地发挥作用。
- 反弹端口型木马。木马开发者在分析了防火墙的特性后发现：防火墙对于连入的连接往往会进行非常严格的过滤，但是对于连出的连接却疏于防范。于是，与一般的木马相反，反弹端口型木马的服务端（被控制端）使用主动端口，客户端（控制端）使用被动端口。木马定时监测控制端的存在，发现控制端上线立即弹出端口主动连接控制端打开的主动端口；为了隐蔽起见，控制端的被动端口一般开在 80，即使用户使用扫描软件检查自己的端口，发现类似 TCP UserIP：3020 ControllerIP:80ESTABLISHED 的情况，稍微疏忽一点，就会以为是自己在浏览网页。

### 1.3.3 拒绝服务攻击和分布式拒绝服务攻击

#### 1. DoS 攻击

DoS 的英文全称是 Denial of Service，也就是“拒绝服务”的意思。从网络攻击的各种方法和所产生的破坏情况来看，DoS 是一种很简单但又很有效的进攻方式。其目的就是拒绝用户的服务访问，破坏服务器的正常运行，最终会使用户的部分 Internet 连接和网络系统失效，如图 1-4 所示。DoS 的攻击方式有很多种，最基本的 DoS 攻击就是利用合理的服务请求来占用过多的服务资源，从而使合法用户无法得到服务。

DoS 攻击的基本过程是：首先攻击者向服务器发送众多的带有虚假地址的请求，服务器发送回复信息后等待回传信息，由于地址是伪造的，所以服务器一直等不到回传的消息，分配给

这次请求的资源就始终没有被释放。当服务器等待一定的时间后，连接会因超时而被切断，攻击者会再度传送新的一批请求，在这种反复发送伪地址请求的情况下，服务器资源最终会被耗尽。

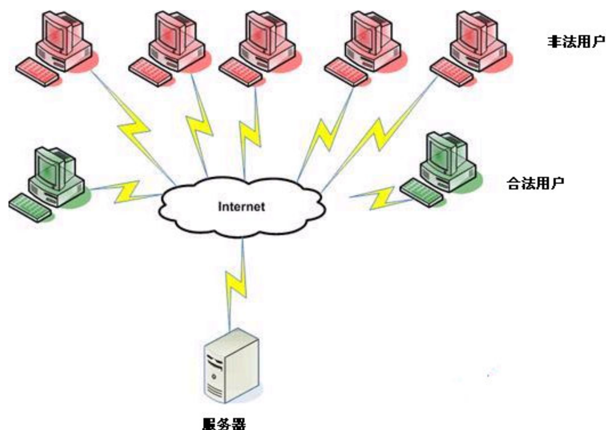


图 1-4　拒绝服务攻击示意图

DoS 根据按照利用漏洞产生的来源划分，可以分为以下几类。

1）利用软件实现的缺陷

OOB 攻击（常用工具 winnuke）、teardrop 攻击（常用工具 teardrop.c boink.c bonk.c）、land 攻击、IGMP 碎片包攻击、jolt 攻击、Cisco 2600 路由器 IOS version 11.0（10）远程拒绝服务攻击等，这些攻击都是利用了被攻击软件实现上的缺陷完成 DoS 攻击的。通常这些攻击工具向被攻击系统发送特定类型的一个或多个报文，这些攻击通常都是致命的，一般都是一击致死，而且很多攻击是可以伪造源地址的，所以即使通过 IDS 或者别的 Sniffer 软件记录到攻击报文也不能找到是谁发动的攻击，而且此类型的攻击多是特定类型的几个报文，非常短暂的少量的报文，如果伪造源 IP 地址的话，几乎是不可能进行追查的。

2）利用协议的漏洞

这种攻击的生存能力非常强。为了能够在网络上进行互通、互联，所有的软件实现都必须遵循既有的协议，而如果这种协议存在漏洞的话，那么所有遵循此协议的软件都会受到影响。

最经典的攻击是 SYN Flooding 攻击，它利用 TCP/IP 协议的漏洞完成攻击。通常一次 TCP 连接的建立包括 3 个步骤，客户端发送 SYN 包给服务器端，服务器分配一定的资源给这里连接并返回 SYN/ACK 包，并等待连接建立的最后的 ACK 包，最后客户端发送 ACK 报文，这样两者之间的连接建立起来，并可以通过连接传送数据了。而攻击的过程就是疯狂发送 SYN 报文，而不返回 ACK 报文，服务器占用过多资源，而导致系统资源占用过多，没有能力响应别的操作，或者不能响应正常的网络请求。这个攻击是经典的以小搏大的攻击，即自己使用少量资源占用对方大量资源。一台 Pentium 4 的 Linux 系统大约能发 30～40M 的 64 字节的 SYN Flooding 报文，而一台普通的服务器 20M 的流量就基本没有任何响应了（包括鼠标、键盘）。而且 SYN Flood 不仅可以远程进行，而且可以伪造源 IP 地址，给追查造成很大困难，要查找必须对所有骨干网络运营商，一级一级路由器地向上查找。

对于伪造源 IP 的 SYN Flooding 攻击，除非攻击者和被攻击的系统之间所有的路由器的管理者都配合查找，否则很难追查。当前一些防火墙产品声称有抗 DoS 的能力，但通常其能力有限，包括国外的硬件防火墙大多 100Mbps 防火墙的抗 SYN Flooding 的能力只有 20～30Mbps（64 字节 SYN 包），这里涉及它们对小报文的转发能力，再大的流量甚至能导致防火墙崩溃。现在有些安全厂商认识到 DoS 攻击的危害，开始研发专用的抗拒绝服务产品。

由于 TCP/IP 协议相信报文的源地址，另一种攻击方式是反射拒绝服务攻击，另外还可以利用广播地址和组播协议辅助反射拒绝服务攻击，这样效果更好。不过大多数路由器都禁止广播地址和组播协议的地址。

还有一类攻击方式是使用大量符合协议的正常服务请求，由于每个请求耗费很大系统资源，导致正常服务请求不能成功。如HTTP协议是无状态协议，攻击者构造大量搜索请求，这些请求耗费大量服务器资源，导致DoS攻击。这种方式的攻击比较好处理，由于是正常请求，暴露了正常的源IP地址，直接禁止这些IP就可以了。

3）资源消耗

这种攻击方式凭借丰富的资源，发送大量的垃圾数据侵占用户的资源，导致DoS。比如，ICMP Flood，mstream Flood，Connection Flood。为了获得比目标系统更多的资源，攻击者通常会发动DDoS（Distributed DoS，分布式拒绝服务攻击），攻击者控制多个攻击“肉鸡”发动攻击，这样才能产生预期的效果。前两类攻击是可以伪造IP地址的，追查也非常困难，第3种攻击由于需要建立连接，可能会暴露攻击“肉鸡”的IP地址，通过防火墙禁止这些IP就可以了。对难于追查、禁止的攻击行为，须要依靠专用的抗拒绝服务产品。

4）常见的DoS攻击类型

在日常的网络威胁中，以下几类DoS攻击较为常见。

- 死亡之ping（ping of death）攻击。ICMP（控制信息协议）在Internet上用于错误处理和传递控制信息。它的功能之一是与主机联系，通过发送一个回应请求（echo request）信息包看看主机是否“存活”。最普通的ping程序就是这个功能。而在TCP/IP的RFC文档中对包的最大尺寸都有严格限制规定，许多操作系统的TCP/IP协议栈都规定ICMP包大小为64KB，且在对包的标题头进行读取之后，要根据该标题头里包含的信息来为有效载荷生成缓冲区。“Ping of Death”攻击就是故意产生畸形的测试Ping（Packet Internet Groper）包，声称自己的尺寸超过ICMP上限，也就是加载的尺寸超过64KB上限，使未采取保护措施的网络系统出现内存分配错误，导致TCP/IP协议栈崩溃，从而造成最终接收方宕机。
- Teardrop攻击。也叫“泪滴攻击”，它利用在TCP/IP协议栈实现中信任IP碎片中的包的标题头所包含的信息来实现自己的攻击。IP分段含有指示该分段所包含的是原包的哪一段的信息，某些TCP/IP协议栈在收到含有重叠偏移的伪造分段时将崩溃。
- UDP“泛洪”攻击（UDP flooding）。在Internet上UDP（用户数据包协议）的应用比较广泛，很多提供WWW和Mail等服务的设备通常是使用Linux服务器，它们默认打开一些被黑客恶意利用的UDP服务。如echo服务会显示接收到的每一个数据包，而原本作为测试功能的chargen服务会在收到每一个数据包时随机反馈一些字符。UDP flood假冒攻击就是利用这两个简单的TCP/IP服务的漏洞进行恶意攻击，通过伪造与某一主机的chargen服务之间的一次的UDP连接，回复地址指向开着echo服务的一台主机，通过将chargen和echo服务互指，来回传送毫无用处且占满带宽的垃圾数据，在两台主机之间生成足够多的无用数据流，这一拒绝服务攻击飞快地导致网络可用带宽耗尽。
- SYN“泛洪”攻击（SYN flooding）。该攻击方式如同上面介绍的利用协议的漏洞中的攻击方式一致，此处不再赘述。
- Land攻击（Land Attack）。在Land攻击中，黑客利用一个特别伪造（spoof）的SYN包（它的原地址和目标地址都被设置成某一个服务器地址）进行攻击。此举将导致接收服务器向它自己的地址发送SYN/ACK消息，结果这个地址又发回ACK消息并创建一个空连接，每一个这样的连接都将保留直到超时，在Land攻击下，许多Linux将崩溃。

- IP 欺骗（IP spoofing）攻击：这种攻击利用 TCP 协议栈的 RST 位来实现，使用 IP 欺骗，迫使服务器把合法用户的连接复位，影响合法用户的连接。假设现在有一个合法用户 A 已经同服务器建立了正常的连接，攻击者构造攻击的 TCP 数据，伪装自己的 IP 为 A 的 IP 地址，并向服务器发送一个带有 RST 位的 TCP 数据段。服务器接收到这样的数据后，认为从用户 A 发送的连接有错误，就会清空缓冲区中已建立好的连接。这时，合法用户 A 再发送合法数据，服务器就已经没有这样的连接了，该用户就被拒绝服务而只能重新开始建立新的连接。

### 2. DDoS 攻击

1）DDoS 攻击原理

与基本的 DoS 攻击不同，DDoS（Distributed Denial of Service，分布式拒绝服务）攻击是一种基于 DoS 的特殊形式的拒绝服务攻击，是一种分布、协作的大规模攻击方式，主要瞄准比较大的站点，像商业公司、搜索引擎和政府部门的站点。通常，DoS 攻击只要一台单机和一个 Modem 就可实现，与之不同的是，DDoS 攻击是利用一批受控制的机器向一台机器发起攻击，这样来势迅猛的攻击令人难以防备，因此具有较大的破坏性。

如图 1-5 所示，一个经典的 DDoS 攻击体系分成四大部分。第一部分是黑客（hacker），它们通常对控制的傀儡机发号施令。第二部分和第三部分分别是控制傀儡机和攻击傀儡机，它们分别是用做控制和实际发起攻击。第四部分则是最终的受害者（被攻击者，victim）。在图 1-5 中需要特别注意控制机与攻击机的区别：对第四部分的受害者来说，DDoS 的实际攻击包是从第三部分攻击傀儡机上发出的，第二部分的控制机只发布命令而不参与实际的攻击。对第二和第三部分计算机，黑客有控制权或者是部分的控制权，并把相应的 DDoS 程序上传到这些平台上，这些程序与正常的程序一样运行并等待来自黑客的指令，通常它还会利用各种手段隐藏自己不被别人发现。在平时，这些傀儡机器并没有什么异常，只是一旦黑客连接到它们进行控制，并发出指令的时候，攻击傀儡机就成为攻击者去发起攻击了。也就是说，其实从严格意义上来说，除了第一部分的黑客之外，第 2～3 部分都是受害者。

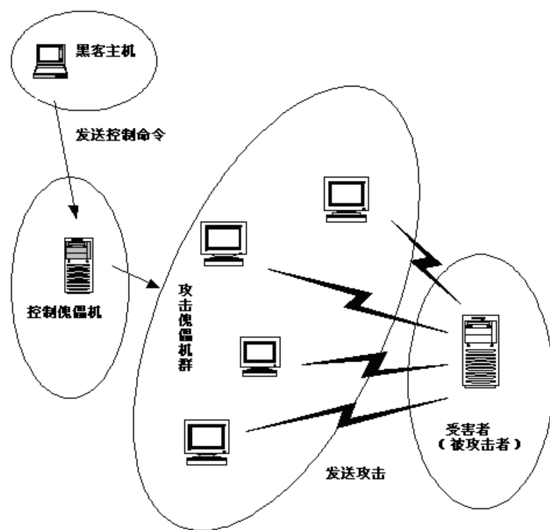


图 1-5　分布式拒绝服务攻击体系结构

2）DDoS 常见的攻击类型

Smurf、Fraggle 攻击、Trinoo、Tribe Flood Network（TFN）、TFN2k 以及 Stacheldraht 是比较常见的 DDoS 攻击程序。

- Smurf 攻击。Smurf 攻击是以最初发动这种攻击的程序名“Smurf”来命名的。它是一种简单但有效的 DDoS 攻击技术，Smurf 还是利用 ping 程序进行源 IP 假冒的直接广播

进行攻击。Smurf 用一个伪造的源地址连续 ping 一个或多个计算机网络，导致所有计算机响应的那个主机地址并不是实际发送这个信息包的攻击计算机。这个伪造的源地址，实际上就是攻击的目标，它将被极大数量的响应信息量所淹没。对这个伪造信息包做出响应的计算机网络就成为攻击的不知情的同谋。一个简单的 Smurf 攻击最终导致网络阻塞和第三方崩溃，这种攻击方式要比 ping of death 洪水的流量高出一两个数量级。这种使用网络发送一个包而引出大量回应的方式也被称为“Smurf 放大”。

- Fraggle 攻击。Fraggle 攻击对 Smurf 攻击作了简单的修改，使用的是 UDP 应答消息而非 ICMP。
- Trinoo 攻击。Trinoo 是较为复杂的 DDoS 攻击程序，它是基于 UDP flooding 的攻击技术。它使用“master”程序对实际实施攻击的任何数量的“代理”程序实现自动控制。当然在攻击之前，侵入者为了安装软件，已经控制了装有 master 程序的计算机和所有装有代理程序的计算机。攻击者连接到安装了 master 程序的计算机，启动 master 程序，然后根据一个 IP 地址的列表，由 master 程序负责启动所有的代理程序。接着，代理程序用 UDP 信息包冲击网络，向被攻击目标主机的随机端口发出全零的 4 字节 UDP 包，在处理这些超出其处理能力垃圾数据包的过程中，被攻击主机的网络性能不断下降，直到不能提供正常服务，乃至崩溃。
- Tribe Flood Network（TFN）和 TFN2K 攻击。Tribe Flood Network 攻击与 Trinoo 攻击一样，使用一个 master 程序与位于多个网络上的攻击代理进行通信，利用 ICMP 给代理服务器下命令，其来源可以做假。TFN 可以并行发动数不胜数的 DoS 攻击，类型多种多样，而且还可建立带有伪装源 IP 地址的信息包。可以由 TFN 发动的攻击包括：SYN flood、UDP flood、ICMP 回应请求 flood 及 Smurf（利用多台服务器发出海量数据包，实施 DoS 攻击）等攻击。TFN 的升级版 TFN2K 进一步对命令数据包加密，更难查询命令内容，命令来源可以做假，还有一个后门控制代理服务器。
- Stacheldraht 攻击。Stacheldraht 攻击同样基于 TFN 和 Trinoo 一样的客户机/服务器（client/server，CS）模式，其中 master 程序与潜在的成千上万个代理程序进行通信。在发动攻击时，侵入者与 master 程序进行连接。Stacheldraht 增加了新的功能：攻击者与 master 程序之间的通信是加密的，对命令来源做假，而且可以防范一些路由器用 RFC2267 过滤，若检查出有过滤现象，它将只做假 IP 地址最后 8 位，从而让用户无法了解到底是哪几个网段的哪台机器被攻击；同时使用远程复制技术对代理程序进行自动更新。Stacheldraht 同 TFN 一样，可以并行发动数不胜数的 DoS 攻击，类型多种多样，而且还可建立带有伪装源 IP 地址的信息包。Stacheldraht 所发动的攻击包括 UDP flooding、TCP SYN flooding 等。

### 1.3.4　病毒

虽然从目前发现的病毒来说，Linux 系统下的病毒相对于 Windows 系统下的来说要少很多，然而却不可对其忽视。本节将介绍 Linux 病毒的发展历程以及 Linux 下病毒的分类。

#### 1. 病毒的起源和历程

1996 年的 Staog 是 Linux 系统下的第一个病毒，它出自澳大利亚一个叫做 VLAD 的组织（Windows 95 下的第一个病毒程序 Boza 也系该组织所为）。Staog 病毒是用汇编语言编写的，专

门感染二进制文件，并通过三种方式去尝试得到 root 权限。Staog 病毒并不会对系统有什么实质性的损坏。该病毒应该算是一个演示版，向世人揭示了 Linux 可能被病毒感染的潜在危险。Linux 系统上第二个被发现的病毒是 Bliss 病毒，它是一个不小心被释放出来的实验性病毒。与其他病毒不同的是，Bliss 本身带有免疫程序，只要在运行该程序时加上 disinfect-files-please 选项，即可恢复系统。

如果说刚开始时 Linux 病毒向人们展示的仅仅是一个概念，那么，在 2001 年发现的 Ramen 病毒，则已经开始引起很多人的担心。Ramen 病毒可以自动传播，无须人工干预，所以和 1988 年曾使人们大受其害的 Morris 蠕虫非常相似。该病毒只感染 Red Hat Linux 6.2 和 7.0 版使用匿名 FTP 服务的服务器，通过两个普通的漏洞 RPC.statd 和 wu-FTP 感染系统。表面看来，这不是一个危险的病毒。它很容易被发现，且不会对服务器做出任何有破坏性的事情。但是当其开始扫描时，将消耗大量的网络带宽。

从 1996 年至今，新的 Linux 病毒屈指可数，这说明 Linux 是一个健壮的具有先天病毒免疫能力的操作系统。当然，出现这种情况，除了其自身设计优秀外，还有其他的原因。首先，Linux 早期的使用者一般都是专业人士，就算是今天，虽然其使用者激增，但典型的使用者仍为那些有着很好的电脑背景且愿意帮助他人的人，Linux 高手更倾向于鼓励新手支持这样一种文化精神。正因为如此，Linux 使用群中一种倾向就是为安全起见尽量避免感染病毒。其次，年轻也是 Linux 很少受到病毒攻击的原因之一。事实上，所有的操作系统（包括 DOS 和 Windows）在其产生之初，也是很少受到各种病毒侵扰的。

然而，2001 年 3 月，美国 SANS 学院的全球事故分析中心 GIAC（Global Incident Analysis Center）发现，一种新的针对使用 Linux 系统的计算机的蠕虫病毒正通过互联网迅速蔓延，它将有可能对用户的电脑系统造成严重破坏。这种蠕虫病毒被命名为“狮子”病毒，与 Ramen 蠕虫病毒非常相似。但是，这种病毒的危险性更大，“狮子”病毒能通过电子邮件把一些密码和配置文件发送到一个位于 china.com 的域名上。事实上，Ramen 病毒是一种比较友善的病毒，它侵入系统后会自动关闭其中的漏洞，而这个病毒却让那些漏洞敞开并开辟新的漏洞。以至于如果用户的系统感染了这个病毒，还不能百分之百确信这个系统有挽救的价值，他们很有可能选择转移其数据并且重新格式化硬盘。

一旦计算机被彻底感染，“狮子”病毒就会强迫电脑开始在互联网上搜寻别的受害者。不过，感染“狮子”病毒的系统少于感染 Ramen 病毒的系统，但是其所造成的损失却比后者大得多。

随着 Klez 病毒在 Linux 平台上的传染，防毒软件厂商开始提醒我们，微软的操作系统不再是惟一易受病毒攻击的操作系统了。即使 Linux 和其他一些主流 UNIX 平台的用户可能不是微软捆绑应用软件的大用户，不可能通过这些软件造成病毒的泛滥，Linux 和 UNIX 仍然有其自身并不引人注目的脆弱点。除了 Klez 以外，其他 Linux/UNIX 平台的主要威胁有：Lion.worm、OSF.8759 病毒、Slapper、Scalper、Linux.Svat 和 BoxPoison 病毒，这些都很少被提及。

病毒的制造者是一些精通编写代码的黑客，他们远比那些胡乱涂改网站却对编写病毒知之甚少的黑客要危险得多。一个被黑掉的网站可以很快修复好，而病毒却更加隐蔽，会带来潜在的安全隐患，它会一直潜伏，直到给系统带来不可挽回的损害。

另外，越多的 Linux 系统连接到局域网和广域网，就会有越多受攻击的可能，这是因为很多 Linux 病毒正在快速地扩散着。使用 WINE 的 Linux/UNIX 系统特别容易受到病毒的攻击。WINE 是一个公开源代码的兼容软件包，能让 Linux 平台运行 Windows 应用软件。WINE 系统特别容易

遭受病毒的攻击，因为它们会使无论是对 Linux 的还是对 Windows 的病毒、蠕虫和木马都能对系统产生威胁。

### 2. 病毒的主要类型

Linux 平台下病毒的分类非常多，主要有以下几类。

- 可执行文件型病毒。可执行文件型病毒是指能够寄生在文件中的、以文件为主要感染对象的病毒。病毒制造者无论使用什么武器，汇编或者 C，要感染 ELF（Executive Linked File，一种为 Linux 系统所采用的通用文件格式，支持动态连接）文件都是轻而易举的事情。这方面的病毒如 Lindose，当其发现一个 ELF 文件时，它将检查被感染的机器类型是否为 Intel 80386，如果是，则查找该文件中是否有一部分的大小大于 2784 字节（或十六进制 AEO），如果满足这些条件，病毒将用自身代码覆盖它并添加宿主文件的相应部分的代码，同时将宿主文件的入口点指向病毒代码部分。一个名为 Alexander Bartolich 的学生发表了一篇名为《如何编写一个 Linux 的病毒》的文章，详细描述了如何制作一个感染在 Linux/i386 的 ELF 可执行文件的寄生文件病毒。有了这样具启发性的、在网上发布的文档，基于 Linux 的病毒数量只会增长得更快。
- 蠕虫（worm）病毒（见图 1-6）。1988 年 Morris 蠕虫爆发后，Eugene H. Spafford 为了区分蠕虫和病毒，给出了蠕虫的技术角度的定义，“计算机蠕虫可以独立运行，并能把自身的一个包含所有功能的版本传播到另外的计算机上。”（worm is a program that can run by itself and can propagate a fully working version of itself to other machines.）。在 Linux 平台下，蠕虫病毒极为猖獗，像利用系统漏洞进行传播的 Ramen、Lion、Slapper 等这些臭名昭著的病毒每一个都感染了大量的 Linux 系统，造成了巨大的损失。它们就是开放源代码世界的 nimda——红色代码。在未来，这种蠕虫病毒仍然会愈演愈烈，Linux 系统应用越广泛，蠕虫的传播程度和破坏能力也会随之增加。

**The Spread of the Sapphire/Slammer Worm**

- The geographic spread of Sapphire/Slammer Worm 30 minutes after release

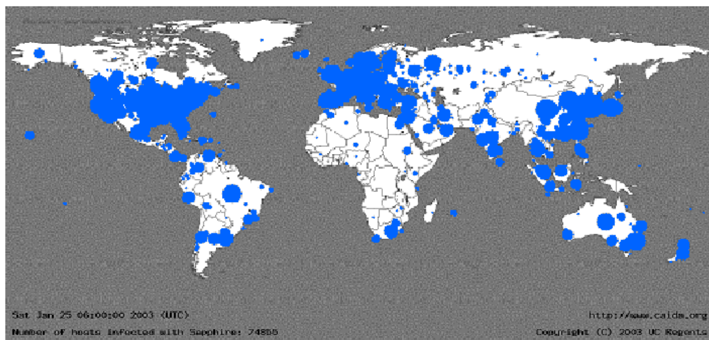



图 1-6　蠕虫病毒传播速度示意

- 脚本病毒。目前出现得比较多的是使用 shell 脚本语言编写的病毒。此类病毒编写较为简单，但是破坏力同样惊人。我们知道，Linux 系统中有许多以.sh 结尾的脚本文件，而一个短短数十行的 shell 脚本就可以在短时间内遍历整个硬盘中所有的脚本文件，进

行感染。因此病毒制造者不要具有很高深的知识，就可以轻易编写出这样的病毒，对系统进行破坏，其破坏性可以是删除文件、破坏系统正常运行，甚至下载一个木马到系统中等。

- 后门程序。在广义的病毒定义概念中，后门也已纳入了病毒的范畴。活跃在 Windows 系统中的后门这一入侵者的利器在 Linux 平台下同样极为活跃。从增加系统超级用户账号的简单后门，到利用系统服务加载、共享库文件注射、rootkit 工具包，甚至可装载内核模块（LKM），Linux 平台下的后门技术发展非常成熟，隐蔽性强，难以清除。是 Linux 系统管理员极为头疼的问题。

从目前的情况来看，由于 Linux 的开源性以及世界各地高手的维护，其下的病毒相对 Windows 来说要少很多，而且从整体上来说，Linux 的抗病毒性从先天上也优于 Windows 操作系统。然而，随着病毒技术的不断发展，由于某些物质利益的驱动，众多的黑客极有可能投身于 Linux 病毒的开发，那么，Windows 下的病毒发展史，也有可能在 Linux 上重演。

### 1.3.5 IP 欺骗

即使是很好地实现了 TCP/IP 协议，但由于它本身存在着一些不安全的地方，从而可以对 TCP/IP 网络进行攻击。这些攻击包括序列号欺骗、路由攻击、源地址欺骗和授权欺骗等。实际上，IP 欺骗（IP Spoofing）不是进攻的结果，而是进攻的手段。进攻实际上是信任关系的破坏。

IP 欺骗由若干步骤组成，这里先简要地描述一下。首先，目标主机已经选定。其次，信任模式已被发现，并找到了一个被目标主机信任的主机。黑客为了进行 IP 欺骗，进行以下工作：使得被信任的主机丧失工作能力，同时采样目标主机发出的 TCP 序列号，猜测出它的数据序列号。然后，伪装成被信任的主机，同时建立起与目标主机基于地址验证的应用连接。如果成功，黑客可以使用一种简单的命令放置一个系统后门，以进行非授权操作。其基本技术原理图如图 1-7 所示。

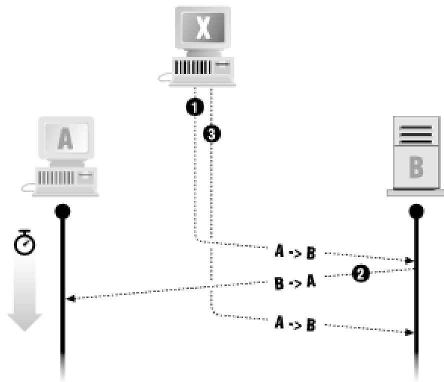



图 1-7　IP 欺骗基本原理示意图

### 1.3.6 ARP 欺骗

如同 IP 欺骗一样，ARP 欺骗（ARP Spoofing）也是黑客常用的攻击手段之一，如图 1-8 所示。ARP 欺骗分为两种，一种是对路由器 ARP 表的欺骗；另一种是对内网 PC 的网关欺骗。

- 路由器 ARP 表欺骗。该欺骗的原理是截获网关数据。它通知路由器一系列错误的内网 MAC 地址，并按照一定的频率不断进行，使真实的地址信息无法通过更新保存在路由器中，结果路由器的所有数据只能发送给错误的 MAC 地址，造成正常计算机无法收到信息，而恶意的计算机则可以通过路由器得到被错误转发的数据包，从而窃取别人的机密信息。

- 网关欺骗。它的原理是建立假网关，让被它欺骗的计算机向假网关发数据，而不是通过正常的路由器途径上网。在计算机看来，就是无法正常上网。

一般来说，ARP 欺骗攻击的后果非常严重，大多数情况下会造成大面积网络掉线和正常用户的数据被错误地发往网络上的恶意计算机。因此，从另外一个侧面来说，它也可以作为一种拒绝服务攻击来理解：正常的用户无法收到数据，而网络无法正常工作。

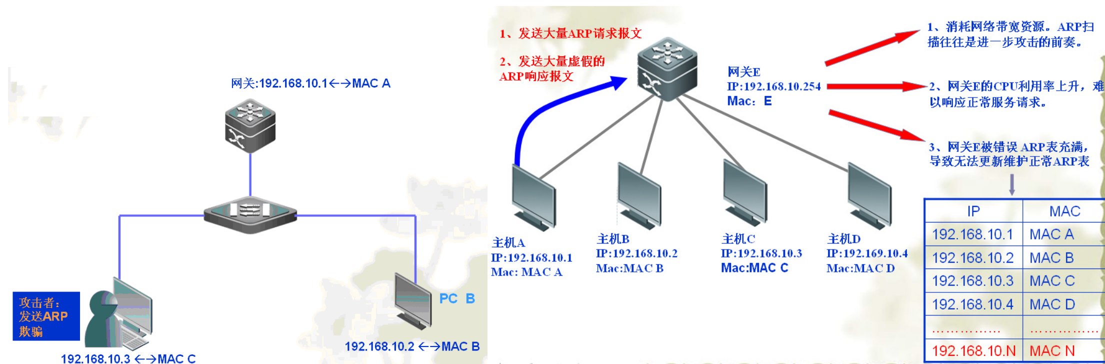



图 1-8　ARP 欺骗示意图

## 1.3.7　网络钓鱼

“网络钓鱼（Phishing）”就其本身来说，称不上是一种独立的攻击手段，更多的只是诈骗方法，和现实中的一些诈骗差不多，如图 1-9 所示。黑客利用欺骗性的电子邮件和假冒的 Web 站点（见图 1-10）来进行诈骗活动，诱骗访问者提供一些个人信息，如信用卡号、账户号和口令、社保编号等内容（通常主要是那些和财务、账号有关的信息）。黑客通常会将自己伪装成知名银行、在线零售商和信用卡公司等可信的品牌单位，因此，受害者往往也是那些和电子商务有关的服务商和使用者。随着 2005 年美国 4000 万信用卡信息被窃案的发生，Phishing 事件受到国内外的密切关注。一些组织估计此类攻击造成的损失超过数百亿美元。国际反网络欺诈组织 APWG 发布的统计报告显示，2004 年 9 月发现活跃的仿冒网站数目为 546 个，12 月则达到 1707 个。CNCERT/CC 在 2004 年共处理 200 多起 Phishing 事件，其中上半年 20 起，8 月以后每个月都接到几十起 Phishing 事件的报告。这类攻击事件越来越频繁，造成的危害也越来越大。

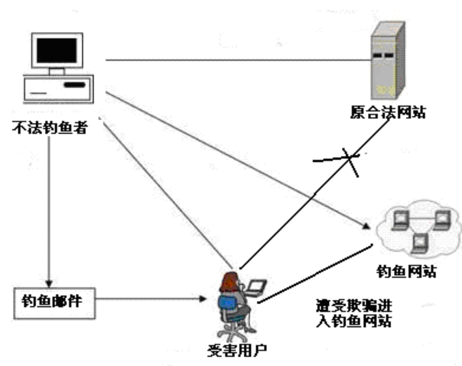



图 1-9　常见的“网络钓鱼”方式示意




图 1-10　网络钓鱼网站图示

网络钓鱼一词是“Fishing”和“Phone”的综合体，由于黑客始祖起初是以电话作案，所以

用“Ph”来取代“F”，创造了“Phishing”。然而，当今的“网络钓鱼”攻击利用欺骗性的电子邮件和伪造的 Web 站点来进行诈骗活动，受骗者往往会泄露自己的财务数据，如信用卡号、账户用户名、口令和社保编号等内容。诈骗者通常会将自己伪装成知名银行、在线零售商和信用卡公司等可信的品牌，在所有接触诈骗信息的用户中，有高达 5%的人都会对这些骗局做出响应。网上黑客采用的“网络钓鱼”方法比较多，归纳起来大概有以下几种方法：

（1）发送垃圾邮件。该类方法以虚假信息引诱用户中圈套，黑客大量发送欺诈性邮件，这些邮件多以中奖、顾问、对账等内容引诱用户在邮件中填入金融账号和密码，或是以各种紧迫的理由（如在某超市或商场刷卡消费，要求用户核对），要求收件人登录某网页提交用户名、密码、身份证号、信用卡号等信息，继而盗窃用户资金。

（2）建立假冒网上银行、网上证券网站。骗取用户账号密码实施盗窃的黑客建立起域名和网页内容都与真正网上银行系统、网上证券交易平台极为相似的网站，诱使用户登录并输入账号密码等信息，进而通过真正的网上银行、网上证券系统盗窃资金，还可利用合法网站服务器程序上的漏洞，在该站点的某些网页中插入恶意 Html 代码，屏蔽那些可用来辨别网站真假的重要信息，利用 cookies 窃取用户信息。

（3）URL 隐藏。根据超文本标记语言的规则可以对文字制作超链接，这样就使网络钓鱼者有机可乘。查看信件源代码就能很快找出其中的奥秘，网络钓鱼者把它写成了：<A HREF=http://www.Abank.com.cn>http://www.Bbank.com.cn</A> ，这样屏幕上就显示了 Bbank 的网址而实际上却链接到了 Abank 的陷阱网站。

（4）利用虚假的电子商务进行诈骗。黑客建立电子商务网站，或是在比较知名、大型的电子商务网站上发布虚假的商品销售信息，黑客在收到受害人的购物汇款后就销声匿迹。除少数黑客自己建立电子商务网站外，大部分黑客采用在知名电子商务网站上，如“易趣”、“淘宝”、“阿里巴巴”等，发布虚假信息，以所谓“超低价”、“免税”、“走私货”、“慈善义卖”的名义出售各种产品，或以次充好，很多人在低价的诱惑下上当受骗。网上交易多是异地交易，通常需要汇款。黑客一般要求消费者先付部分款，再以各种理由诱骗消费者付余款或者其他各种名目的款项，得到钱款或被识破时，就立即切断与消费者的联系。

（5）利用木马和黑客技术窃取用户信息后实施盗窃。黑客通过发送邮件或在网站中隐藏木马等方式大肆传播木马程序，当感染木马的用户进行网上交易时，木马程序可获取用户账号和密码，并发送给指定邮箱，用户资金将受到严重威胁。

（6）利用用户弱口令等漏洞破解、猜测用户账号和密码。黑客利用部分用户密码设置得过于简单的账号，对账号密码进行破解。目前已有很多的弱口令破解黑客工具在网上可以免费下载，它们可以在很短的时间内破解出各类比较简单的用户名及密码。

（7）其他手段。实际上，黑客在实施“网络钓鱼”犯罪活动的过程中，经常采取以上几种手法交织、配合进行。并不排除有新的手段的出现，比如现今泛滥成灾的“垃圾手机短信”，其中有部分是诈骗短信，以急迫的口吻要求用户对并不存在的已消费的“商品”进行买单，提供账户和密码，严格地说，它也应当属于“网络钓鱼”的范畴。所以，推而广之，任何通过网络手段（包括通信）进行诈骗和误导用户使之遭受经济损失的行为都应当称之为“网络钓鱼”。

个人用户要避免成为 Phishing 的受害者，一定要加强安全防范意识，提高安全防范技术水平。针对性的措施可以归纳为以下几点。

（1）防范垃圾邮件。这是防范网络钓鱼最为重要和关键的一步。当今绝大部分的垃圾邮件都

携带有网络钓鱼的链接，用户经常收到莫名其妙的邮件，因为好奇而点击其中的链接，随之而来的便是被其中的“廉价”或者“伪冒”信息所蛊惑，或者是被安装上木马。因此，利用垃圾邮件防护工具或者主动地对不明邮件提高警惕是防范网络钓鱼的第一要义。

（2）安装防病毒系统和网络防火墙系统。这是一个非常必须的步骤，多数反病毒软件都具有对包括间谍软件、木马程序的查杀功能；防火墙系统监视着系统的网络连接，能够杜绝部分攻击意图并及时报警提醒用户注意。由于病毒和黑客攻击手段翻新不断，防病毒和防火墙系统应及时升级，定期杀毒。

（3）及时给操作系统和应用系统打补丁，堵住软件漏洞。像 Windows 操作系统和 IE 浏览器软件都存在很多已知和未知的漏洞，一般厂家在发现漏洞之后会迅速推出相应的补丁程序，用户应当经常跟踪操作系统和应用程序的官方网站，充分利用厂商的资源，在发现各种漏洞时第一时间为自己的系统打上安全补丁，避免黑客利用漏洞入侵电脑，减少潜在威胁。

（4）从主观意识上提高警惕性，提高自身的安全技术。首先要注意核对网址的真实性，在访问重要的网站时最好能记住其网络域名或者 IP 地址，确保登录到正确的网站，避免点击搜索引擎搜索出的链接等简便方法。第二要养成良好的使用习惯，不要轻易登录访问陌生网站、黄色网站和有黑客嫌疑的网站，拒绝下载安装不明来历的软件，拒绝可疑的邮件，及时退出交易程序，做好交易记录及时核对等。

（5）妥善保管个人信息资料。很多银行为了保障用户的安全，设定了登录密码（查询密码）和支付密码（取款密码）两套密码，用户若保证登录密码与支付密码不相同，这样即使登录密码被窃取，网络钓鱼者依然无法操作用户的资金。尽量选择安全的密码，建议选用字母、数字混合的方式，以提高密码猜测和破解的难度。密码等个人资料应妥善保管并定期更新，避免将密码泄露给他人。

（6）采用新的安全技术。数字证书是一种很安全的方式，通过数字证书可以进行安全通信和电子数字签名，电子签名具有法律效力。网上交易在数字证书签名和加密的保护下进行网上数据的传送，杜绝了网络钓鱼者使用跨站 cookie 攻击以及嗅探侦测的可能。数字证书具有可复制性，如同家门钥匙一样，用户应妥善保管。对于一些被假冒的机构和政府相关管理部门而言，也应采取相应的措施与 Phishing 这种犯罪活动做斗争。例如，银行也可积极采取技术措施和宣传活动让用户能够识别真假，避免上当。相关政府职能部门也应沟通合作，及时定位、关闭这些仿冒网站并从其所有者手中追回被盗的用户信息，减少直接和潜在损失。当然，实际工作中会面临很多困难，如技术上的困难和司法方面的困难。国家计算机网络应急技术处理协调中心 CNCERT/CC 在 2004 年处理的 Phishing 事件，基本上都是利用境外网站实施的，难以及时关闭。

### 1.3.8 僵尸网络

僵尸网络，英文名称为 Botnet，是 20 世纪 90 年代末兴起的危害互联网的产物，近年来已形成重大安全威胁之一。僵尸网络是指采用一种或多种传播手段，将大量主机感染 bot 程序（僵尸程序），从而在控制者和被感染主机之间所形成的一个可一对多控制的网络。

攻击者通过各种途径传播僵尸程序，感染互联网上的大量主机，而被感染的主机将通过一个控制信道接收攻击者的指令，组成一个僵尸网络。之所以用僵尸网络这个名字，是为了更形象地让人们认识到这类危害的特点：众多的计算机在不知不觉中如同中国古老传说中的僵尸群一样被人驱赶和指挥着，成为被人利用的一种工具。

僵尸网络可以说是一个可控制的网络，这个网络并不是指物理意义上具有拓扑架构的网络，它具有一定的分布性，随着 bot 程序的传播而不断有新位置的僵尸计算机添加到这个网络中来。这个网络是采用了一定的恶意传播手段形成的，例如，主动漏洞攻击、邮件病毒等各种病毒与蠕虫的传播手段，都可以用来进行 Botnet 的传播，从这个意义来讲，恶意程序 bot 也是一种病毒或蠕虫，如图 1-11 所示。

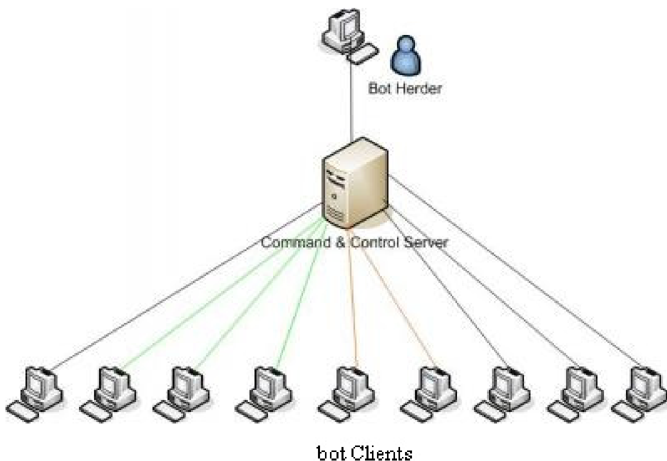



图 1-11　僵尸网络示意图

Botnet 最主要的特点是，它有别于以往简单的安全事件，是一个具有极大危害的攻击平台。它可以一对多地执行相同的恶意行为，将攻击源从一个转化为多个，乃至一个庞大的网络体系，通过网络来控制受感染的系统，同时从异地造成网络危害，比如可以同时对某目标网站进行 DDoS 攻击，同时发送大量的垃圾邮件，短时间内窃取大量敏感信息、抢占系统资源进行非法牟利等。

僵尸网络正是这种一对多的控制关系，使得攻击者能够以极低的代价高效地控制大量的资源为其服务，这也是 Botnet 攻击模式近年来受到黑客青睐的根本原因。在执行恶意行为的时候，Botnet 充当了一个攻击平台的角色，这也就使得 Botnet 不同于简单的病毒和蠕虫，也与通常意义的木马有所不同。

僵尸网络这种受控网络的存在，给危害追踪和损失抑制带来巨大的麻烦。这也就是僵尸网络迅速发展的原因。僵尸网络已经成为国内乃至全世界的网络安全领域最为关注的危害之一。

### 1.3.9　跨站脚本攻击

跨站脚本（Cross-Site Scripting，XSS）攻击（见图 1-12）指的是恶意攻击者往 Web 页面里插入恶意 html 代码，当用户浏览该页面时，嵌入 Web 里面的 html 代码会被执行，从而达到恶意用户的特殊目的。XSS 属于被动式的攻击，因为其被动且不好利用，所以许多人常忽视了其危害性。

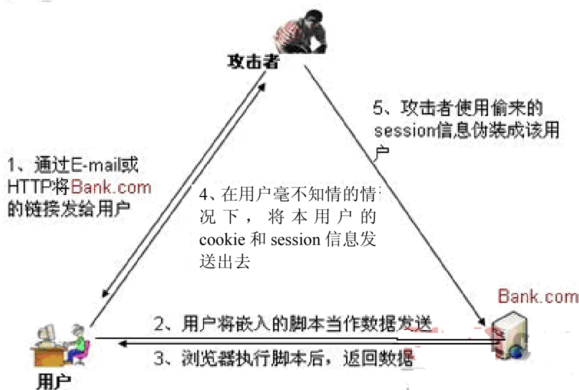



图1-12　跨站脚本攻击示意图

XSS 攻击分成两类，一类是来自内部的攻击，主要指的是利用程序自身的漏洞，构造跨站语句；另一类则是来自外部的攻击，主要指的是自己构造 XSS 跨站漏洞网页或者寻找非目标机以外的有跨站漏洞的网页。

XSS 攻击有多种方式，主要方式是入侵者通过技术手段在某个页面里插入一个恶意 html 代码。当然，攻击者有时也会在网页中加入一些以.JS 或.VBS 为后缀名的代码时，在用户浏览时，就会被攻击到。目前，以下类型的脚本可以被插入到远程页面中，以完成 XSS 攻击。

- HTML
- Java
- VB
- ActiveX
- Flash

## 1.3.10　缓冲区溢出攻击

缓冲区溢出是一种非常普遍和危险的漏洞，在各种操作系统、应用软件中广泛存在。利用缓冲区溢出攻击，可以导致程序运行失败、系统宕机、重新启动等后果。更为严重的是，可以利用它执行非授权指令，甚至可以取得系统特权，进而进行各种非法操作。

缓冲区溢出（见图 1-13）是指当计算机向缓冲区内填充数据位数时超过了缓冲区本身的容量溢出的数据覆盖在合法数据上，理想的情况是程序检查数据长度并不允许输入超过缓冲区长度的字符，但是绝大多数程序都会假设数据长度总是与所分配的储存空间相匹配，这就为缓冲区溢出埋下隐患。操作系统所使用的缓冲区又被称为“堆栈”。在各个操作进程之间，指令会被临时储存在“堆栈”当中，“堆栈”也会出现缓冲区溢出。

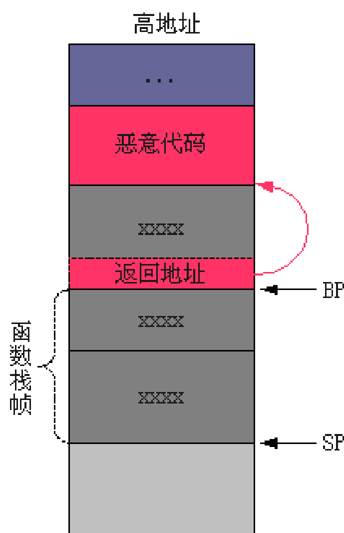


图1-13　缓冲区溢出攻击示意

在当前网络与分布式系统安全中，被广泛利用的 50%以上都是缓冲区溢出，其中最著名的例子是 1988 年利用 fingerd 漏洞的蠕虫。而缓冲区溢出中，最为危险的是堆栈溢出，因为入侵者可以利用堆栈溢出，在函数返回时改变返回程序的地址，让其跳转到任意地址，带来的其中一种危害是程序崩溃导致拒绝服务，另外一种危害就是跳转并且执行一段恶意代码，比如得到 shell，然后为所欲为。

历史上最著名的缓冲区溢出攻击可能要算是1988年11月2日的Morris Worm所携带的攻击代码了。这个因特网蠕虫利用fingerd程序的缓冲区溢出漏洞，给用户带来了极大危害。此后，越来越多的缓冲区溢出漏洞被发现。从bind、wu-ftpd、telnetd、apache等常用服务程序，到Microsoft、Oracle等软件厂商提供的应用程序，都存在着似乎永远也弥补不完的缓冲区溢出漏洞。

## 1.3.11　SQL 注入攻击

SQL 注入攻击是黑客对数据库进行攻击的常用手段之一。随着 B/S 模式应用开发的发展，使用这种模式编写应用程序的程序员也越来越多。但是由于程序员的水平及经验也参差不齐，相当大一部分程序员在编写代码的时候，没有对用户输入数据的合法性进行判断，使应用程序存在安全隐患。用户可以提交一段数据库查询代码，根据程序返回的结果，获得某些他想得知的数据，

这就是所谓的 SQL Injection，即 SQL 注入。

SQL 注入是从正常的 WWW 端口访问，而且表面看起来和一般的 Web 页面访问没什么区别，所以目前市面的防火墙都不会对 SQL 注入发出警报，如果管理员没查看 Web Server 日志的习惯，可能被入侵很长时间都不会发觉。但是，SQL 注入的手法相当灵活，在注入的时候会碰到很多意外的情况，需要构造巧妙的 SQL 语句，从而成功获取想要的数据，如图 1-14 所示。




图1-14　SQL注入攻击示意图

### 1.3.12　零日攻击

与上述攻击方式相比，零日攻击并不指代一种特定的攻击技术，它更多的是指代一种快速采用软件系统等的漏洞对目标进行攻击的过程和现象。

据权威机构统计，现在我们使用的操作系统和应用程序，每 1000 行代码中就有可能存在 4～5 个编码漏洞。由于系统和应用程序的开发商不可能对所有的代码进行严格的检查，一般都是在系统和应用程序发行后，不断地进行后期测试才会不断发现这些漏洞，然后才会开发相应的漏洞补丁来修补这些漏洞，这就是我们经常说的系统补丁更新。但是，当系统或应用程序发行后，一些技术高超的黑客也会对它们进行反汇编，然后通过阅读代码，来找到编码中的漏洞。而由黑客找到的漏洞，他们并不会对外公布，最多也只是在其熟悉的内部流传。然后黑客就会自己编写此漏洞的利用脚本，对系统或应用程序发动攻击。通常，从系统或应用程序新漏洞的发现到利用，不会超过 24 小时，由此人们就形象地把这种攻击称为“零日攻击”，将被利用的新漏洞称为“零日漏洞”。

通常，黑客实施一次零日攻击会按以下流程来进行。

- 发现漏洞。
- 编写漏洞利用脚本。
- 寻找目标发动攻击。
- 进入目标网络或系统。
- 安装后门控制系统或得到网络中机密数据。
- 清除攻击痕迹。

但是，并不是所有的黑客都会按这样的流程来进行零日攻击，他们可能在进入系统或控制系统后，会对系统所在网络中的其他目标发动新的攻击，以得到更多的数据；或者将被攻入的系统作为攻击其他网络目标的跳板。

目前，所有的操作系统，包括 Windows、类 Linux、FreeBSD 和 UNIX，以及所有的网络应用程序，包括 IIS、FTP、IE 和 SMTP 等都存在零日漏洞，也就存在被零日攻击的风险。正是由于零日攻击影响的范围非常广泛，而且，由于操作系统或应用程序在编码过程中不可能做到百分之百的没有错误，因此，零日攻击的风险会在以后相当长的一段时间里都会存在，因而现在我们就必须了解可以通过什么样的方法来解决零日攻击带来的安全风险。

## 1.3.13 “社会工程学 ”攻击

社会工程学（Social Engineering)是一种利用人的弱点：如人的本能反应、好奇心、信任、贪婪等进行诸如欺骗、伤害等危害手段，获取自身利益的手法。近年来，由于信息安全厂商不断开发出更先进的安全产品，系统安全防范在技术上越来越严密，使得攻击者利用技术上的漏洞变得越来越困难。于是，更多的人转向利用人为因素的手段——社会工程学来进行攻击。

许多信息技术从业者都普遍存在着类似的一种观念：他们认为自己的系统部署了先进、周密的安全设备，包括防火墙、IDS、IPS、漏洞扫描、防病毒网关、内容过滤、安全审计、身份认证和访问控制系统，甚至于最新的UTM和防水墙，以为靠这些安全设施即可保证系统的安全。事实上，很多安全行为出现在骗取内部人员（信息系统管理、使用、维护人员等）的信任，从而轻松绕过所有技术上的保护。信任是一切安全的基础，对于保护与审核的信任，通常被认为是整个安全链条中最薄弱的一环。为规避安全风险，技术专家精心设计的安全解决方案，却很少重视和解决最大的安全漏洞，那就是人为因素。因此，对于当前的企业来说，缺乏对社会工程学防范的信息系统，不管其安全技术多么先进完善，很可能会成为一种自我安慰的摆设，其投入大笔资金购置的最先进的安全设备，很可能成为一种浪费。

社会工程学攻击基本上可以分为两个层次：物理的和心理的。与以往的入侵行为相类似，社会工程学在实施之前要完成很多相关的前期工作，这些工作甚至要比后续的入侵行为本身更为繁重和更具技巧。这些工作包括：社会工程学的实施者（一般称为社会工程师）必须掌握心理学、人际关系学、行为学等知识与技能，以便收集和掌握实施入侵行为所需要的相关资料与信息。通常为了达到预期目的，社会工程学攻击都要将心理的和行为的攻击两者结合运用。其常见形式包括以下几种。

第一，伪装。从早期的求职信病毒、爱虫病毒、圣诞节贺卡到目前流行的网络钓鱼，都是利用电子邮件和伪造的 Web 站点来进行诈骗活动。有调查显示，在所有接触诈骗信息的用户中，有高达 5%的人都会对这些骗局做出响应。攻击者越来越喜欢玩弄社会工程学的手段，把恶件、间谍软件、勒索软件（ransom-ware）、流氓软件等网络陷阱伪装起来欺骗被害者。

第二，引诱。社会工程学是现在多数蠕虫病毒进行传播时所使用的技术，它使计算机用户本能地去打开邮件，执行具有诱惑性同时又具有危害性的附件。例如，用一些关于某些型号的处理器存在运算瑕疵的“瑕疵声明”或更能引起人的兴趣的“幸运中奖”、“最新反病毒软件”等说辞，并给出一个页面链接，诱惑你进入该页面运行下载程序或在线注册个人相关信息，利用人们疏于防范的心理引诱你上钩。

第三，恐吓。利用人们对安全、漏洞、病毒、木马、黑客等内容会特别敏感，以权威机构的面目出现，散布诸如安全警告、系统风险之类的信息，使用危言耸听的伎俩恐吓和欺骗计算机用户，声称如果不及时按照他们的要求去做就会造成致命的危害或遭受严重损失。

第四，说服。社会工程师说服目标的目的是增强他们主动完成所指派的任务的顺从意识，从而成为一个可以被信任并由此获得敏感信息的人。大多数企业咨询帮助台人员一般接受的训练都是要求他（她）们热情待人并尽可能地为来人或来电提供帮助，所以这里就成了社会工程学实施者获取有价值信息的“金矿”。

第五，恭维。社会工程师通常十分友善，很讲究说话的艺术，知道如何借助机会去迎合人，投其所好，使多数人会友善地作出回应，恭维和虚荣心的对接会让目标乐意继续合作。

第六，渗透。通常社会工程学攻击者都擅长刺探信息，很多表面上看起来毫无用处的信息都会被他们利用来进行系统渗透。通过观察目标对电子邮件的响应速度、重视程度以及可能提供的相关资料，比如一个人的姓名、生日、ID、电话号码、管理员的 IP 地址、邮箱等都可能被利用起来，通过这些收集信息来判断目标的网络架构或系统密码的大致内容，从而用口令心理学来分析口令，而不仅仅是使用暴力破解。

除了以上的攻击手段，一些比较另类的行为也开始在社会工程学中出现，其中包括像翻垃圾（dumpster diving）、背后偷窥（shoulder surfing）、反向社会工程学等都是窃取信息的捷径。

### 1.3.14　中间人攻击

中间人攻击（Man-in-the-MiddleAttack，简称“MITM 攻击”）是一种“间接”的入侵攻击，这种中间人攻击是通过各种技术手段将受入侵者控制的一台计算机虚拟放置在网络连接中的两台通信计算机之间，这台计算机就称为“中间人”。然后入侵者把这台计算机模拟成一台或两台原始计算机，使“中间人”能够与原始计算机建立活动连接并允许其读取或修改传递的信息，然而两个原始计算机用户却认为他们是在互相通信。通常，这种“拦截数据——修改数据——发送数据”的过程就被称为“会话劫持（SessionHijack）”。

曾经猖獗一时的 SMB 会话劫持、DNS 欺骗等技术都是典型的 MITM 攻击手段。在黑客技术越来越多地运用于以获取经济利益为目标的情况下时，MITM 攻击成为对网银、网游、网上交易等最有威胁并且最具破坏性的一种攻击方式。

### 1.3.15　密码攻击

密码攻击是指在不知道密钥的情况下，恢复出明文。通常情况下，是指黑客通过手上掌握的相关情况和信息，来对应用系统等所设立密码的猜测和破解。主要包括以下几类。

- 唯密文攻击：攻击者仅能获得一些加密过的密文。
- 已知明文攻击：攻击者有一些密文并且知道相对应的明文。
- 选择明文攻击：攻击者在开始攻击之前可以选择一些明文并从系统中获得相对应的密文。如果攻击者在攻击中途可以根据已经获得的信息选择新的明文并获得对应的密文，则称为适应性选择明文攻击。
- 选择密文攻击：攻击者在开始攻击之前可以选择一些密文并从系统中获得相对应的明文。如果攻击者在攻击中途可以根据已经获得的信息选择新的密文并获得对应的明文，则称为适应性选择密文攻击。
- 相关钥匙攻击：与选择明文（或密文）攻击类似。不同的是，攻击者可以得到被两个不同的钥匙所加密（或解密）得到的密文（或明文）。攻击者不知道这两个钥匙的数值，但知道这两个钥匙之间的关系，比如两个钥匙之间相差一个比特。

## 1.4　认识黑客

黑客（英语：Hacker，或称骇客），是指对计算机科学、编程和设计方面具有高度理解的人。Hacker 这一词有以下几种不同的意思。

- 在信息安全（Information security）业里，“黑客”指研究智取计算机安全系统的人员。包括利用公共通信网路，如互联网和电话系统，在未经许可的情况下，进入对方系统的黑帽黑客（英语：black hat，另称 Cracker），也包括调试和分析计算机安全系统的白帽黑客（英语：white hat）。最早用来称呼研究盗用电话系统的人士。
- 在业余计算机方面，“黑客”指研究修改计算机产品的业余爱好者。20 世纪 70 年代，很多的这些群落聚焦在硬件研究，20 世纪 80 年代和 90 年代，很多的群落聚焦在软件更改（如编写游戏模组、攻克软件版权限制）。
- 依照 RFC 1392，“黑客”是“一位热衷于研究系统和计算机（特别是计算机网络）内部运作秘密的人”。根据此宽容的定义，黑客也可以包括很多计算机和互联网技术创造者。

为了避免误解，公众都必须明确地区分开 Hacker 与 cracker 的概念：“黑客”和“骇客”当中的中文音译“黑”或“骇”字总使人对黑客有所误解，真实的黑客主要指的是高级程序员，如 Linux 创始人林纳斯•托瓦兹，而不是为人所误解的专指对电脑系统及程序进行恶意攻击及破坏的人。除了精通编程，精通操作系统如 UNIX 的人可以被视作黑客外，现在精通网络入侵的人也被看作是“黑客”，但一般被称为骇客，对硬件设备创新的工程师通常也被认为是黑客。

“黑客”（Hacker）一词，一般有以下意义。

- 一个对（某领域内的）编程语言有足够了解，可以不经长时间思考就能创造出有用的软件的人。
- 喜爱编程（Coding）并享受在其中，变得更擅长于编程的人。
- 喜爱自由（Freedom），不易受约束，但觉得假如是为了喜爱的事物，可以受适当的约束。

一个试图破解某系统或网络以提醒该系统所有者的电脑安全漏洞，这群人往往被称作“白帽黑客”或“思匿客（sneaker）”。许多这样的人是电脑安全公司的雇员，并在完全合法的情况下攻击某系统。Hacker 是一个通过知识或猜测而对某段程序做出（往往是好的）修改，并改变（或增强）该程序用途的人。

“骇客”（Cracker）一词一般有以下意义：一个恶意（一般是非法地）试图破解或破坏某个程序、系统及网络安全的人。在使用简体字地区，这群人应称为“骇客”，但于正体字使用时称黑客、怪客、垮客和剑客（以读音直译，此处与简体字地区的用词有所出入）等。

- “Cracker”不同于“Hacker”。
- “Cracker”没有“Hacker”精神，也没有道德标准。
- “Hacker”们建设，而“Cracker”们破坏。

而为了与目前业界的称谓进行统一，本书中我们所谈到的黑客就特指 Cracker（符合上述的第一种含义），指的是对信息系统、网络或者数据实施破坏、窃听等行为的人。

## 1.5 剖析黑客的攻击手段

黑客进行网络攻击是一件系统性很强的工作，其主要工作流程可以分为三大阶段：攻击前准备、攻击实施和攻击后处理。又可以细分为以下七步：确定攻击目标、踩点和信息收集、获得权

限、权限提升、攻击实施、留取后门程序、掩盖入侵痕迹，如图 1-15 所示。

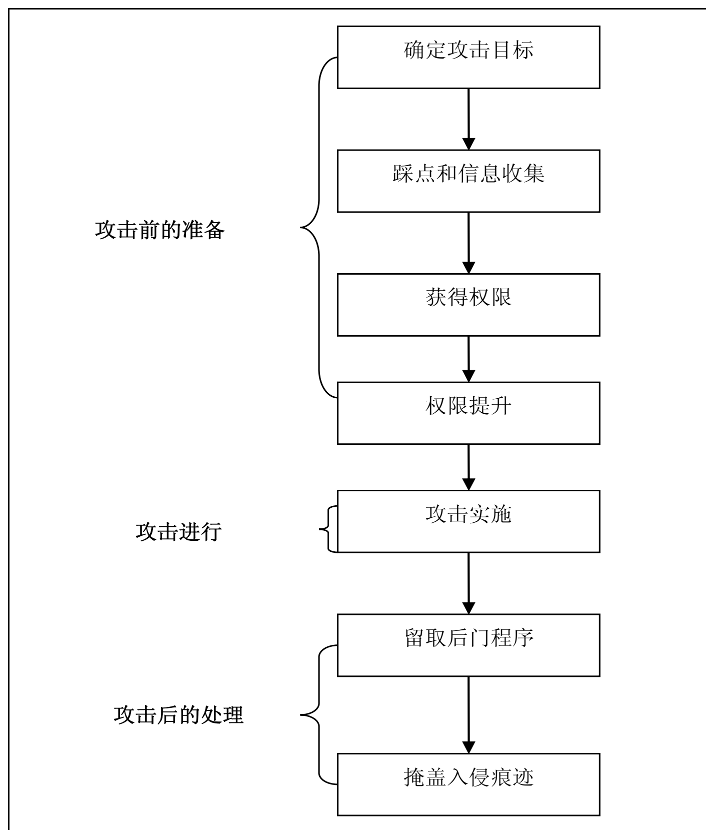



图 1-15　黑客攻击步骤示意图

## 1.5.1　确定攻击目标

攻击者在进行一次完整的攻击之前首先要确定攻击要达到什么样的目的，即给对方造成什么样的后果。常见的攻击目的有破坏型和入侵型两种。破坏型攻击指的只是破坏攻击目标，使其不能正常工作，而不能随意控制目标的系统的运行。要达到破坏型攻击的目的，主要的手段是拒绝服务攻击（Denial of Service）。另一类常见的攻击目的是入侵攻击目标，这种攻击是要获得一定的权限来达到控制攻击目标的目的。应该说这种攻击比破坏型攻击更为普遍，威胁性也更大。因为黑客一旦获取攻击目标的管理员权限就可以对此服务器做任意动作，包括破坏性的攻击。此类攻击一般也是利用服务器操作系统、应用软件或者网络协议存在的漏洞进行的。当然还有另一种造成此种攻击的原因就是密码泄露，攻击者靠猜测或者穷举法来得到服务器用户的密码，然后就可以用和真正的管理员一样的权限对服务器进行访问。

## 1.5.2　踩点和信息搜集

除了确定攻击目的之外，攻击前的最主要工作就是通过“踩点”，收集尽量多的关于攻击目标的信息。这些信息主要包括目标的操作系统类型及版本，目标提供哪些服务，各服务器程序的类型与版本以及相关的社会信息。

要攻击一台机器，首先要确定它上面正在运行的操作系统是什么，因为对于不同类型的操作

系统，其上的系统漏洞有很大区别，所以攻击的方法也完全不同，甚至同一种操作系统的不同版本的系统漏洞也是不一样的。要确定一台服务器的操作系统一般是靠经验，有些服务器的某些服务显示信息会泄露其操作系统。还有一种不是很有效的方法，诸如查询 DNS 的主机信息（不是很可靠）来看登记域名时的申请机器类型和操作系统类型，或者使用社会工程学的方法来获得，以及利用某些主机开放的 SNMP 的公共组来查询。

另外一种相对比较准确的方法是利用网络操作系统里的 TCP/IP 堆栈作为特殊的“指纹”来确定系统的真正身份。因为不同的操作系统在网络底层协议的各种实现细节上略有不同。可以通过远程向目标发送特殊的包，然后通过返回的包来确定操作系统类型。例如通过向目标机发送一个 FIN 的包（或者是任何没有 ACK 或 SYN 标记的包）到目标主机的一个开放的端口，然后等待回应。许多系统如 Windows、 BSDI、Cisco、HP / UX 和 IRIX 会返回一个 RESET。通过发送一个 SYN 包，它含有没有定义的 TCP 标记的 TCP 头。那么在 Linux 系统的回应包就会包含这个没有定义的标记，而在一些别的系统则会在收到 SYN+BOGU 包之后关闭连接。或是利用寻找初始化序列长度模板与特定的操作系统相匹配的方法。利用它可以对许多系统分类，如较早的 UNIX 系统是 64KB 长度，一些新的 UNIX 系统的长度则是随机增长。还有就是检查返回包里包含的窗口长度，这项技术根据各个操作系统的不同的初始化窗口大小来惟一确定它们。利用这种技术实现的工具很多，比较著名的有 NMAP、CHECKOS、QUESO 等。

获知目标提供哪些服务及各服务 Daemon 的类型、版本同样非常重要，因为已知的漏洞一般都是对某一服务的。这里说的提供服务就是指通常我们提到的端口，例如一般 telnet 在 23 端口，FTP 在 21 端口，WWW 在 80 端口或 8080 端口，这只是一般情况，网站管理完全可以按自己的意愿修改服务所监听的端口号。在不同服务器上提供同一种服务的软件也可以是不同的，我们管这种软件叫做 Daemon，例如同样是提供 FTP 服务，可以使用 wuftp、proftp，ncftp 等许多不同种类的 Daemon。确定 Daemon 的类型版本也有助于黑客利用系统漏洞攻破网站。

另外需要获得的关于系统的信息就是一些与计算机本身没有关系的社会信息，例如网站所属公司的名称、规模，网络管理员的生活习惯、电话号码等。这些信息看起来与攻击一个网站没有关系，实际上很多黑客都是利用了这类信息攻破网站的。例如有些网站管理员用自己的电话号码做系统密码，如果掌握了该电话号码，就等于掌握了管理员权限。进行信息收集可以用手工进行，也可以利用工具来完成，完成信息收集的工具叫做扫描器。用扫描器收集信息的优点是速度快，可以一次对多个目标进行扫描。

### 1.5.3 获得权限

当收集到足够的信息之后，攻击者就要开始实施攻击行动了。作为破坏性攻击，只需利用工具发动攻击即可。而作为入侵性攻击，往往要利用收集到的信息，找到其系统漏洞，然后利用该漏洞获取一定的权限。有时获得了一般用户的权限就足以达到修改主页等目的了，但作为一次完整的攻击是要获得系统最高权限的，这不仅是为了达到一定的目的，更重要的是证明攻击者的能力，这也符合黑客的追求。

能够被攻击者所利用的漏洞不仅包括系统软件设计上的安全漏洞，也包括由于管理配置不当而造成的漏洞。前不久，因特网上应用最普及的著名 WWW 服务器提供商 Apache 的主页被黑客攻破，其主页面上的 Powered by Apache 图样（羽毛状的图画）被改成了 Powered by Microsoft Backoffice 的图样，那个攻击者就是利用了管理员对 Webserver 数据库的一些不当配置而成功取

得最高权限的。

当然大多数攻击成功的范例还是利用了系统软件本身的漏洞。造成软件漏洞的主要原因在于编制该软件的程序员缺乏安全意识。当攻击者对软件进行非正常的调用请求时造成缓冲区溢出或者对文件的非法访问。其中利用缓冲区溢出进行的攻击最为普遍，据统计 80%以上成功的攻击都是利用了缓冲区溢出漏洞来获得非法权限的。

无论作为一个黑客还是一个网络管理员，都需要掌握尽量多的系统漏洞。黑客需要用它来完成攻击，而管理员需要根据不同的漏洞来进行不同的防御措施。了解最新最多的漏洞信息，可以到诸如 Rootshell（www.rootshell.com）、Packetstorm（packetstorm.securify.com）、Securityfocus（www.securityfocus.com）等网站去查找。

### 1.5.4 权限提升

系统漏洞分为远程漏洞和本地漏洞两种。远程漏洞是指黑客可以在别的机器上直接利用该漏洞进行攻击并获取一定的权限。这种漏洞的威胁性相当大，黑客的攻击一般都是从远程漏洞开始的。但是利用远程漏洞获取的不一定是最高权限，而往往只是一个普通用户的权限，这样常常没有办法做黑客想要做的事。这时就需要配合本地漏洞来把获得的权限进行扩大，常常是扩大至系统的管理员权限。只有获得了最高的管理员权限之后，才可以做诸如网络监听、打扫痕迹之类的事情。要完成权限的扩大，不但可以利用已获得的权限在系统上执行利用本地漏洞的程序，还可以放一些木马之类的欺骗程序来套取管理员密码，这种木马是放在本地套取最高权限用的，而不能进行远程控制。例如一个黑客已经在一台机器上获得了一个普通用户的账号和登录权限，那么他就可以在这台机器上放置一个假的 su 程序。一旦黑客放置了假 su 程序，当真正的合法用户登录时，运行了 su，并输入了密码，这时 root 密码就会被记录下来，下次黑客再登录时就可以使用 su 变成 root 了。

### 1.5.5 攻击实施

在获得能够进行攻击和破坏的权限后，攻击者就可以为所欲为地对目标进行攻击了，包括篡改重要文件、信息窃取、系统破坏、上传程序等工作。

### 1.5.6 留取后门程序

一般黑客都会在攻入系统后不止一次地进入该系统。为了下次再进入系统时方便一点，黑客会留下一个后门，否则本次攻击过程在下一次攻击中的工作就白费了，而其中特洛伊木马就是后门的最好范例。Linux 和 UNIX 中留后门的方法有很多种，多达十余种，读者和网络管理员可以查阅相关的工具书籍进行了解和熟悉。

### 1.5.7 掩盖入侵痕迹

如果攻击者完成攻击后就立刻离开系统而不做任何善后工作，那么他的行踪将很快被系统管理员发现，因为所有的网络操作系统一般都提供日志记录功能，会把系统上发生的动作记录下来。所以，为了自身的隐蔽性，黑客一般都会抹掉自己在日志中留下的痕迹。根据操作系统的不同略有变化，攻击者在获得系统最高管理员权限之后就可以随意修改系统上的文件了（只对常规 UNIX

系统而言），包括日志文件，所以一般黑客想要隐藏自己踪迹的话，就会对日志进行修改。最简单的方法当然就是删除日志文件了，但这样做虽然避免了系统管理员根据 IP 追踪到自己，但也明确无误地告诉管理员，系统已经被入侵了。所以最常用的办法是只对日志文件中有关自己的那一部分做修改。关于修改方法的具体细节根据不同的操作系统有所区别，网络上有许多此类功能的程序，例如 zap、wipe 等，其主要做法就是清除 utmp、wtmp、Lastlog 和 Pacct 等日志文件中某一用户的信息，使得当使用 w、who、last 等命令查看日志文件时，隐藏掉此用户的信息。

只修改日志是不够的，因为百密必有一漏，即使自认为修改了所有的日志，仍然会留下一些蛛丝马迹的。例如安装了某些后门程序，运行后也可能被管理员发现。所以，黑客高手可以通过替换一些系统程序的方法来进一步隐藏踪迹。这种用来替换正常系统程序的黑客程序叫做 rootkit，这类程序在一些黑客网站可以找到，比较常见的有 LinuxRootKit。它可以替换系统的 ls、ps、netstat、inetd 等一系列重要的系统程序，当替换了 ls 后，就可以隐藏指定的文件，使得管理员在使用 ls 命令时无法看到这些文件，从而达到隐藏自己的目的。

# 第2章

# 知己：企业信息安全技术概览

## 本章导读

本书第1章介绍了知彼的要素，包括企业信息安全面临的威胁以及黑客惯用的攻击手段和步骤等，那么，摆在企业面前的课题是：如何应对这些安全威胁？作为企业用户及其管理员来说，这是非常重要的。对于企业自身来说，自己有什么技术和手段可以用来抵御黑客入侵，做好信息安全工作呢？

本章将分门别类地详细介绍如何在各个层次来使用信息安全的相关技术来应对安全威胁。

## 2.1 物理层防护：物理隔离

企业物理层面临着对计算机网络与计算机系统的物理装备的威胁，主要表现在自然灾害、电磁辐射与恶劣工作环境方面。而相应的防范措施包括抗干扰系统、物理隔离、防辐射系统、隐身系统。在众多技术当中，物理隔离是最为关键和重要的企业物理层安全防护技术。

在互联网高度发展的今天，访问没有好的防护措施的企业的内部局域网对于外部的恶意访问者来说并非难事，他们通常可以窥探、非法获取甚至是恶意毁坏企业的宝贵数据和资料，从而对企业造成不可挽回的经济损失。当前绝大多数的企业都在物理层面采用了物理隔离的措施将本地局域网和互联网进行隔离，保证两个网络的不可互访，从而达到保护内网数据安全的目的。

目前看来，对安全要求比较高的企业，一般会采用物理隔离网闸来实现企业的网络管理。物理隔离网闸是使用带有多种控制功能的固态开关读写介质连接两个独立主机系统的信息安全设备。由于物理隔离网闸所连接的两个独立主机系统之间，不存在通信的物理连接、逻辑连接、信息传输命令、信息传输协议，不存在依据协议的信息包转发，只有数据文件的无协议“摆渡”，且对固态存储介质只有“读”和“写”两个命令。所以，物理隔离网闸从物理上隔离、阻断了具有潜在攻击可能的一切连接，使“黑客”无法入侵、无法攻击、无法破坏；物理隔离网闸可以解决以下威胁：操作系统漏洞、入侵、基于 TCP/IP 漏洞的攻击、基于协议漏洞的攻击、木马、基于隧道的攻击、基于文件的攻击等。这些是互联网目前存在的绝大部分主要威胁，物理隔离网闸在理论上完全实现了真正的安全。

物理隔离网闸最早出现在美国、以色列、俄罗斯等国家的军方，用以解决涉密网络与公共网络连接时的安全。俄罗斯的 Ry Jones，以色列的 Buky Carmeli、Elad Baron、Daniel Steiner 等人都是该领域的先驱，后者现在美国开公司。目前最大的物理隔离公司当属美国的 Whale communications 公司，Spearhead 公司也不小。随着我国电子政务的快速发展，外部网络连接着广大民众，内部网络连接着政府公务员桌面办公系统，专网连接着各级政府的信息系统，在外网、内网、专网之间交换信息是基本要求。如何在保证内网和专网资源安全的前提下，实现从民众到政府的网络畅通、资源共享、方便快捷是电子政务系统建设中必须解决的技术问题。一般采取的方法是在内网与外网之间实行防火墙的逻辑隔离，在内网与专网之间实行物理隔离。出于对重要信息、文件保护的目的，物理隔离网闸开始被人们接受，物理隔离网闸成为电子政务信息系统必须配置的设备。

物理隔离网闸主要由内网处理单元、外网处理单元、安全隔离与信息交换处理单元三部分组成。外网处理单元与外网（如 Internet）相连；内网处理单元与内网（如军队网）相连；安全隔离与信息交换处理单元通过专用硬件断开内、外网的物理连接，并在任何时刻只与其中一个网络连接，读取等待发送的数据，然后“推送”到另一个网络上。在切换速度非常快的情况下，可以实现信息的实时交换。

物理隔离卡是物理隔离的低级实现形式，一个物理隔离卡只能管一台个人计算机，甚至只可能在 Windows 环境下工作，每次切换都需要开关机一次。物理隔离网闸是物理隔离的高级实现形式，网闸可以管理整个网络，不需要开关机。网闸实现后，原则上不再需要物理隔离卡。安全隔离是一种逻辑隔离，防火墙就是一种逻辑隔离，因此防火墙也是一种安全隔离。有些厂商对安全隔离增加了一些特点，如采用了双主机结构，但双主机之间却是通过包来转发的，如图 2-1 所示。

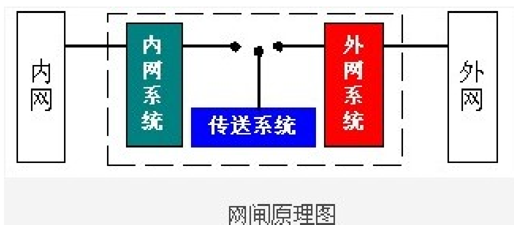


图 2-1　网闸原理图

无论双主机之间采用了多么严格的安全检查，但只要是包转发，就存在基于包的安全漏洞，存在对包的攻击。这在本质上同两个防火墙串联并无本质的差别。从目前已经存在的安全隔离网闸，包括以下类型：通过串口或并口来实现双主机之间的包转发，通过 USB 或 1394 或 firewire（火线）等方式来实现双主机之间的包转发，甚至是直接通过以太线来实现双主机之间的包转发，以及其他任何形式的通信方式来实现双主机之间的包转发如专用 ASIC 开关电路、ATM、Myrinet 卡等，都是安全隔离网闸，但都不是物理隔离网闸。

对于企业用户来说，物理隔离网闸适用于以下五种典型应用场合。

### 1．局域网与互联网之间（内网与外网之间）

有些局域网络，特别是政府办公网络，涉及政府敏感信息，有时需要与互联网在物理上断开，用物理隔离网闸是一个常用的办法。

### 2．办公网与业务网之间

由于办公网络与业务网络的信息敏感程度不同，例如，银行的办公网络和银行业务网络就是很典型的信息敏感程度不同的两类网络。为了提高工作效率，办公网络有时需要与业务网络交换信息。为解决业务网络的安全，比较好的办法就是在办公网与业务网之间使用物理隔离网闸，实现两类网络的物理隔离。

### 3．电子政务的内网与专网之间

在电子政务系统建设中要求政府内网与外网之间用逻辑隔离，在政府专网与内网之间用物理隔离。现常用的方法是用物理隔离网闸来实现。

### 4．业务网与互联网之间

电子商务网络一边连接着业务网络服务器，一边通过互联网连接着广大民众。为了保障业务网络服务器的安全，在业务网与互联网之间应实现物理隔离。

### 5．涉密网与非涉密网之间

电子政务建设中一般都对网络按照安全级别进行了安全域的划分，这在一定程度上保证了信息的安全，非涉密的系统及面向公众的信息采集和发布系统主要运行在非涉密网部分。涉密网、非涉密网之间物理隔离，依照涉密信息“最小化”原则，进行涉密网和非涉密网之间两个不同的信息安全域信息的适度“可靠交换”。

物理隔离网闸在网络的第七层将数据还原为原始数据文件，然后以“摆渡文件”的形式来传递原始数据。任何形式的数据包、信息传输命令和 TCP/IP 协议都不可能穿透物理隔离网闸。根据我国计算机网络安全管理的规定，物理隔离网闸需要取得公安部、国家保密局和中国信息安全测评认证中心的安全产品的测评认证证书。在此基础上，进入军事领域，还需要军队测评认证中心的认证证书。

涉及国家秘密的计算机信息系统，不得直接或间接地与国际互联网或其他公共信息网络相连接，必须实行物理隔离。涉密网络必须在物理隔离的基础上实现WWW浏览和自由收发E-mail。各级政府机关和涉密单位，必须将已建成的办公局域网同Internet或上一级专网实行物理隔离，正在建设的电子政务网络必须实现物理隔离。除了党政军外，其他行业和部门纷纷颁布建设规范和管理办法，要求使用物理隔离。电力、铁道、金融、银行、证券、保险、税务、海关、水利、交通、民航、社保、石化等行业部门，要求在物理隔离的条件下实现安全的数据库数据交换。为了保证电子政务和电子商务体系中关键系统的安全性，物理隔离网闸将确保关键网络和涉密网络万无一失，同时又能提供网络业务的互联互通。

目前市场上出现专门应用于特定行业的物理隔离网闸，如供应电力专用物理隔离网闸是一种专门为电力系统“二次安全防护”设计的一款单向物理隔离网闸。主要应用于实时控制区（I区，或者DCS网）、非控制生产区（II区）与生产管理区（III、IV区，或MIS网）完全的网络物理隔离，并保证I、II区可向III、IV区有效实时地传输数据，但反过来，任何网络入侵、病毒的攻击均被有效地阻隔，这样就可在最大限度上防止黑客、病毒的侵害。

## 2.2 系统层防护：安全操作系统和数据库安全

众所周知，操作系统以及数据库是现代计算机技术中最为底层和核心的软件系统。那么，企业系统层次的安全问题主要来自于网络内使用的操作系统的安全和数据库安全层面上。针对操作系统和数据库的安全技术林林总总，也体现在应用的不同层面上，本节将从保障企业系统层安全的核心层次出发，着重从安全操作系统、密码设定策略及数据库安全技术三方面进行讲解，其他的网络层面和应用层面与其相关的防护技术将在后续专题中详细讲解。

### 2.2.1 选用安全操作系统

操作系统作为计算机系统的最为基础和重要的基础软件和系统软件，起着管理计算机资源，直接利用计算机硬件为用户提供与硬件相关、与应用无关的使用和编程接口的作用。它是上层应用软件获得高可靠性以及信息的完整性、保密性的基础。没有操作系统安全的支持，就没有获得网络安全的可能。为了保证企业计算机网络和信息的安全，有必要采用具有较高安全级别的安全操作系统来作为企业稳固的安全操作平台。美国国防部（DoD）于1983年推出了著名的TCSEC（Trusted Computer System Evaluation Criteria，可信计算机评估准则），也称为“橙皮书”，是迄今为止评估计算机安全最有名的一个标准。它将计算机信息系统的安全分为四等八个级别，由低到高依次为：D、C1、C2、B1、B2、B3、A1、超 A1。按照这个标准，传统的操作系统，包括UNIX操作系统、Windows操作系统大都处于C级，它们的安全级别不是很高。而许多安全级别高的操作系统，比如Trusted Information Systems、Trusted XENIX、SNS等都处于实验室阶段，尚未投入市场。另外，国内对安全操作系统的研究工作也在如火如荼地进行，我国从20世纪90年代开始研究安全操作系统，到目前已经取得了一些成果。从1993年我国宣布开发成功具有B2集功能的操作系统，现在已有众多的公司和科研单位开始注重安全操作系统的研发，比如中科红旗公司开发的红旗 Linux、南京大学的 Softos、国防科大的麒麟安全操作系统、中科安胜公司的安胜操作系统等。

在安全操作系统中，对操作系统进行安全加固和安全理论模型是从根本上解决操作系统安全性问题的两项非常重要的技术。我们首先谈谈操作系统安全加固技术，它有以下几种方法，如图 2-2 所示。

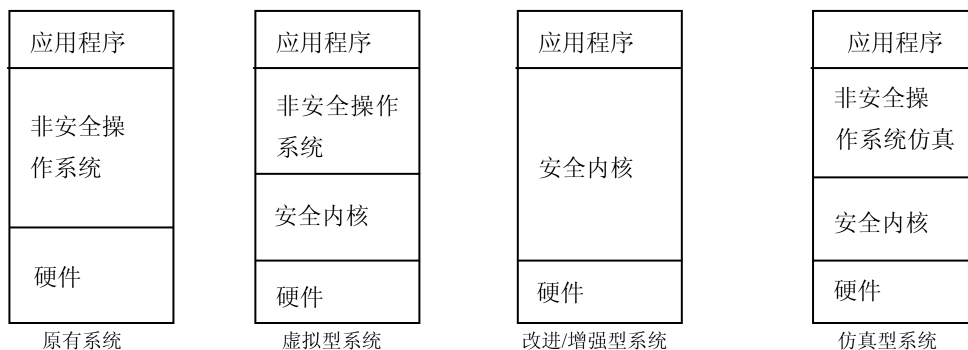



图 2-2　操作系统的几种加固方式

（1）虚拟机法。在现有的操作系统与硬件之间增加一个新的层次，作为安全内核，现有操作系统几乎可以不作任何变动而成为虚拟机。安全内核的接口基本与原有硬件等价，操作系统本身透明地在安全内核的控制之下，它可以不变地支持现有的应用程序，并且能够很好地兼容非安全操作系统的未来版本。硬件特性对虚拟机的实现非常关键，它要求原系统的硬件和结构都必须要支持虚拟机，因而这种加固方式的局限性比较大。

（2）增强法。在现有的操作系统基础上，对原有的内核以及应用程序采用一定的安全策略进行改进，并且加入相应的安全机制，形成新的操作系统内核。这样，增强后的操作系统保持了原来操作系统的用户接口。这种方法是在已有操作系统基础上进行的，不是对原有内核的完全改变，因而受到原有体系结构的限制，难以达到很高的安全级别。但是这种方法并不会破坏原有系统的体系结构，开发代价小，不会影响原系统的运行效率，现在普遍采用的是这种方式。

（3）仿真法。对现有的操作系统的内核作面向安全策略的修改，然后在安全内核与原非安全操作系统的用户接口中嵌入仿真程序。那么，我们在建立安全内核的时候，可以不必如第二种方法一样受到现有应用程序的限制，而且可以完全自由地定义仿真程序与安全内核之间的接口。然而，采用这种方式需要同时设计仿真程序以及安全内核，还要受到原操作系统接口的限制，同时开发难度大，接口复杂。

在安全操作系统的理论模型方面，我们通常提到 BLP 模型。Bell & LaPadula（BLP）模型是由 David Bell 和 Leonard La Padula 于 1973 年提出的安全模型，它同时也是一个状态机模型。它是定义多级安全性的基础，被视作基本安全公理。该公理最早是在 Multics 安全操作系统中得到实施。虽然从商业角度来看，该操作系统并不成功，但是单从安全性角度来看，Multics 迈出了安全操作系统设计的第一步，为后来的安全操作系统的研制工作积累了大量丰富的经验。随后它又在 Secure Xenix、System V/MLS、Tunis、VAX 的 VMM 安全核心等多种安全系统的研制中得到了广泛的应用。

该模型将计算机信息系统中的实体分为两部分，主体和客体。凡是实施操作的称为主体，比如说用户和进程；而被操作的对象则称为客体，比如说文件、数据库等。对主体和客体而言，存在着两种最重要的安全控制方法，它们是强制存取控制以及自主存取控制方式。

- 强制存取控制。主要是通过“安全级”来进行，安全级包含“密级”和“部门级”两

方面，密级又分为无密、秘密、机密、绝密四级。为了实施强制型安全控制，主体和客体均被赋予“安全级”。两者的关系包含三点：① 主体的安全级高于客体，当且仅当主体的密级高于客体的密级，且主体的部门级包含客体的部门级；②主体可以读客体，当且仅当主体安全级高于或者等于客体；③主体可以写客体，当且仅当主体安全级低于或者等于客体。

- 自主存取控制。主体对其拥有的客体，有权决定自己和他人对该客体应具有怎样的访问权限。

在这个模型当中，每个主体有最大安全级和当前安全级，每个客体有一个安全级。主体对客体有四种存取方式：只读、只写、执行以及读写。该模型当中有以下两条很重要的规则，如图2-3所示。

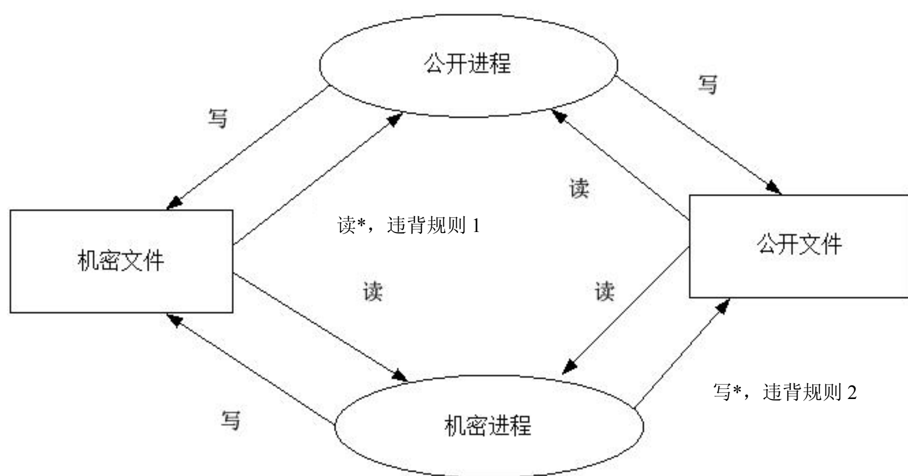



图 2-3　BLP 模型安全规则示意图

（1）简单安全特性。是指主体只能够从下读、向上写。为了使主体对客体既能读又能写，二者的安全级别必须完全一样。

（2）*——特性。主体对客体有“只写”权限，则客体安全级至少和主体的当前安全级一样高；主体对客体有“读”权限，则客体安全级应该小于或者等于主体当前安全级；主体对客体有“读写”权限，则客体安全级等于主体的当前安全级。

## 2.2.2　操作系统密码设定

设定登录密码是一项非常重要的安全措施，如果用户的密码设定得不合适，就很容易被破译，尤其是拥有超级用户使用权限的用户，如果没有良好的密码，将给系统造成很大的安全漏洞。在多用户系统中，如果强迫每个用户选择不易猜出的密码，将会大大提高系统的安全性。但如果passwd 程序无法强迫每个上机用户使用恰当的密码，要确保密码的安全度，就只能依靠密码破解程序了。

实际上，密码破解程序是黑客工具箱中的一种工具，它将常用的密码或者是英文字典中所有可能用来作密码的字都用程序加密成密码字，然后将其与 Linux 系统（/etc/passwd）或者 Windows 系统的密码文件（SAM）相比较，如果发现有吻合的密码，就可以求得明码了。

在网络上可以找到很多密码破解程序，比较有名的程序是 crack。用户可以自己先执行密码破解程序，找出容易被黑客破解的密码，先行改正总比被黑客破解要有利。

目前密码破解程序大多采用字典攻击以及暴力攻击手段，而其中用户密码设定不当，则极易受到字典攻击的威胁。很多用户喜欢用自己的英文名、生日或者账户等信息来设定密码，这样，黑客可能通过字典攻击或者是社会工程手段来破解密码。所以建议用户在设定密码的过程中，应尽量使用非字典中出现的组合字符，并且采用数字与字符相结合、大小写相结合的密码设置方式，增加密码被黑客破解的难度。而且，也可以使用定期修改密码、使密码定期作废的方式，来保护自己的登录密码。

下面具体列出几条参考原则（几个需要遵循）。

- 口令长度至少为 8 个字符。口令越长越好。若使用 MD5 口令，它应该至少有 15 个字符。若使用 DES 口令，则使用最长长度（8 个字符）。
- 混和大小写字母。Linux 区分大小写，因此混和大小写会增加口令的强健程度。
- 混和字母和数字。在口令中添加数字，特别是在中间添加（不只在开头和结尾处）能够加强口令的强健性。
- 包括字母和数字以外的字符。&、$和 > 之类的特殊字符可以极大地增强口令的强健性（若使用 DES 口令，则不能使用此类字符）。
- 挑选一个你可以记住的口令。如果你记不住自己的口令，那么它再好也没有用；使用简写或其他记忆方法来帮助你记忆口令。

另外，还有一些原则需要牢记（几个不要）。

- 不要只使用单词或数字。绝不要在口令中只使用单词或数字。
- 不要使用现成词汇。像名称、词典中的词汇，甚至电视剧或小说中的用语，即使在两端使用数字，都应该避免。
- 不要使用外语中的词汇。口令破译程序经常使用多种语言的词典来检查其词汇列表。依赖外语来达到保护口令的目的通常不起作用。
- 不要使用黑客术语。
- 不要使用个人信息。千万不要使用个人信息。如果攻击者知道你的身份，推导出你所用口令的任务就会变得非常容易。
- 不要倒转现存词汇。优秀的口令破译者总是倒转常用词汇，因此倒转薄弱口令并不会使它更安全。
- 不要笔录你的口令。绝不要把口令写在纸上。把它牢记在心才更为安全。
- 不要在所有机器上都使用同样的口令。在每个机器上使用不同的口令是极其重要的。这样，如果一个系统泄密了，所有其他系统都不会立即受到威胁。

### 2.2.3 数据库安全技术

企业数据库存储着许多敏感性数据，例如，企业机密资料、公司的客户信息资料等。黑客利用操作系统和数据库的漏洞，能轻易地获取你数据库中的数据（经营策略、客户资料、设计资料、人事资料等），这些文件一旦被潜在的黑客或竞争对手窃取，必将给国家或企业带来极大的危害。而且这种危害的严重性在于它是潜在的，根本无法察觉。不像现实社会中物品的失窃，很快就能察觉到，而数据、电子信息一旦被窃，是很难发现的，并且不会留下任何痕迹。也许你的机密资

料已经不知不觉地被窃取，落入竞争对手或不法分子的手中，你却浑然不知，数据的失密将对国家的安全和企业的竞争力产生潜在的、无法估量的威胁。针对这种情况，可以从以下几方面进行安全防护。

1）数据库用户安全管理

企业数据库是一个极为复杂的系统，因此很难进行正确的配置和安全维护，当然，首先必须要保证的就是数据库用户权限的安全性。在企业门户网站中，当用户通过 Web 方式对数据库中的对象（表、视图、触发器、存储过程等）进行操作时，必须通过数据库访问的身份认证。多数数据库系统还有众所周知的默认账号和密码，可支持对数据库资源的各级访问。因此，很多重要的数据库系统很可能受到威胁。用户存取权限是指不同的用户对于不同的数据对象有不同的操作权限。存取权限由两个要素组成：数据对象和操作类型。定义一个用户的存取权限就是要定义该用户可以在哪些数据对象上进行哪些类型的操作。权限分系统权限和对象权限两种。系统权限由 DBA 授予某些数据库用户，只有得到系统权限，才能成为数据库用户。对象权限是授予数据库用户对某些数据对象进行某些操作的权限，它既可由 DBA 授权，也可由数据对象的创建者授予。

2）严格定义用户访问视图

为不同的用户定义不同的视图，可以限制用户的访问范围。通过视图机制把需要保密的数据对无权存取这些数据的用户隐藏起来，可以对数据库提供一定程度的安全保护。实际应用中常将视图机制与授权机制结合起来，首先用视图机制屏蔽一部分保密数据，然后在视图上进一步进行授权。

3）采用数据加密

数据安全隐患无处不在。一些机密数据库、商业数据等必须防止他人非法访问、修改、拷贝。为了保证数据的安全，数据加密是应用最广、成本最低廉而相对最可靠的方法。数据加密是保护数据在存储和传递过程中不被窃取或修改的有效手段。数据加密系统包括对系统的不同部分要选择何种加密算法、需要多高的安全级别、各算法之间如何协作等因素。在系统的不同部分要综合考虑执行效率与安全性之间的平衡。一般来讲，安全性总是以牺牲系统效率为代价的。如果要在 Internet 上的两个客户端传递安全数据，则要求客户端之间可以彼此判断对方的身份，传递的数据必须加密，当数据在传输中被更改时可以被发觉。

4）完善的事务管理和故障恢复机制

事务管理和故障恢复主要是对付系统内发生的自然因素故障，保证数据和事务的一致性和完整性。故障恢复的主要措施是进行日志记录和数据复制。在网络数据库系统中，分布事务首先要分解为多个子事务到各个站点上去执行，各个服务器之间还必须采取合理的算法进行分布式并发控制和提交，以保证事务的完整性。事务运行的每一步结果都记录在系统日志文件中，并且对重要数据进行复制，发生故障时根据日志文件利用数据副本准确地完成事务的恢复。

5）作好数据库备份与恢复

计算机同其他设备一样，都可能发生故障。计算机故障的原因多种多样，包括磁盘故障、电源故障、软件故障、灾害故障以及人为破坏等。一旦发生这种情况，就可能造成数据库的数据丢失。因此数据库系统必须采取必要的措施，以保证发生故障时，可以恢复数据库。数据库管理系统的备份和恢复机制就是保证在数据库系统发生故障时，能够将数据库系统还原到正常状态。加强数据备份非常重要，数据库拥有很多关键的数据，这些数据一旦遭到破坏后果将不堪设想，而这往往是入侵者真正关心的东西。不少管理员在这点上作得并不好，不是备份不完全，就是备份不及时。数据备份需要周密计划，制定出一个策略测试后再去实施，备份计划也需要不断地调整。

6）设定审计追踪机制

审计追踪机制是指系统设置相应的日志记录，特别是对数据更新、删除、修改的记录，以便日后查证。日志记录的内容可以包括操作人员的名称、使用的密码、用户的 IP 地址、登录时间、操作内容等。若发现系统的数据遭到破坏，可以根据日志记录追究责任，或者从日志记录中判断密码是否被盗，以便修改密码，重新分配权限，确保系统的安全。

7）重点做好服务器防护

在网络时代，我们所谈论的企业数据库的安全主要体现在 Web 数据库的层面上。我们知道，在 Web 数据库的三层体系结构中，数据存放在数据库服务器中，大部分的事务处理及商业逻辑处理在应用服务器中进行，由应用服务器提出对数据库的操作请求。理论上，既可以通过 Web 页面调用业务处理程序来访问数据库，也可以绕过业务处理程序，使用一些数据库客户端工具直接登录数据库服务器，存取操作其中的数据。所以，数据库服务器的安全设置至关重要。用 IDS（入侵检测系统）保卫数据库安全逐步普及，这种安全技术将传统的网络和操作系统级入侵检测系统概念应用于数据库。应用 IDS 提供主动的、针对 SQL 的保护和监视，可以保护预先包装或自行开发的 Web 应用。

8）做好数据库升级和打补丁工作

在当今的信息时代，作为与操作系统有着同等重要地位的系统软件，数据库的升级和打补丁工作也是一项长期而不容被企业忽视的工作。随着数据库管理系统功能设计的日益复杂和全面化，其内在的漏洞也不断地暴露出来。据赛门铁克今年的统计数据表明，在漏洞攻击中，记录到 Oracle 数据库 168 个漏洞，其在主要数据库中居然位居第一，充分反映了当前数据库在安全方面所表现出来的薄弱性和受关注程度。幸运的是，当前的各大数据库厂商都有一支专门的队伍在对数据库可能出现的漏洞和问题在做升级和补丁工作，企业的数据库管理员则需要及时地从数据库厂商和网上获得最新的数据库升级和补丁程序，对现有的系统进行更新和升级，从而保证在瞬息万变的网络世界中，企业数据库能够以较好的安全性能来应对各种攻击和挑战。

## 2.3 网络层防护：防火墙

### 2.3.1 防火墙简介

防火墙（Fire Wall）指的是一个由软件和硬件设备组合而成、在内部网和外部网之间、专用网与公共网之间的界面上构造的保护屏障。它是一种获取安全性方法的形象说法，是一种计算机硬件和软件的结合，使 Internet 与 Intranet 之间建立起一个安全网关（Security Gateway），从而保护内部网免受非法用户的入侵。

网络防火墙借鉴了古代真正用于防火的防火墙的喻义，指的是设置在不同网络（如可信任的企业内部网和不可信的公共网）或网络安全域之间的一系列部件的组合。它可通过监测、限制、更改跨越防火墙的数据流，尽可能地对外部屏蔽网络内部的信息、结构和运行状况，以此来实现网络的安全保护。在逻辑上，防火墙是一个分离器、限制器，也是一个分析器，有效地监控了内部网和 Internet 之间的任何活动，保证内部网络的安全。目前防火墙已经开始被企业用户普遍接受，而且正在成为企业网络中一种主要的安全设备。

典型的防火墙体系网络结构（见图2-4）是：防火墙的一端连接企事业单位内部的局域网，而另一端则连接互联网。所有的内、外部网络之间的通信都要经过防火墙。只有符合安全策略的数据流才能通过防火墙，这是防火墙的工作原理特性。防火墙之所以能保护企业内部网络，就是依据这样的工作原理或者说是防护机制进行的。它可以由管理员自由设置企业内部网络的安全策略，使允许的通信不受影响，而不允许的通信全部被拒绝于内部网络之外。

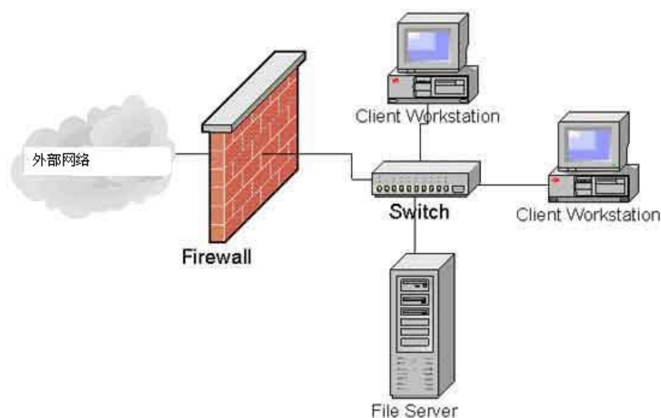



图2-4　防火墙体系网络结构

防火墙对流经它的网络通信进行扫描，这样能够过滤掉一些攻击，以免其在目标计算机上被执行。防火墙还可以关闭不使用的端口。而且它还能禁止特定端口的流出通信，封锁特洛伊木马。最后，它可以禁止来自特殊站点的访问，从而防止来自不明入侵者的所有通信。

- 防火墙是网络安全的屏障。一个防火墙（作为阻塞点、控制点）能极大地提高一个内部网络的安全性，并通过过滤不安全的服务而降低风险。由于只有经过精心选择的应用协议才能通过防火墙，所以网络环境变得更安全。如防火墙可以禁止诸如众所周知的不安全的NFS（Network File System，网络文件系统）协议进出受保护网络，这样外部的攻击者就不可能利用这些脆弱的协议来攻击内部网络。同时防火墙可以保护网络免受基于路由的攻击，如IP选项中的源路由攻击和ICMP重定向中的重定向路径。防火墙应该可以拒绝所有以上类型攻击的报文并通知防火墙管理员。
- 防火墙可以强化网络安全策略。通过以防火墙为中心的安全方案配置，能将所有安全软件（如口令、加密、身份认证、审计等）配置在防火墙上。与将网络安全问题分散到各个主机上相比，防火墙的集中安全管理更经济。例如，在网络访问时，一次一密口令系统和其他的身份认证系统完全可以不必分散在各个主机上，而集中于防火墙一身。
- 对网络存取和访问进行监控审计。如果所有的访问都经过防火墙，那么，防火墙就能记录下这些访问并进行日志记录，同时也能提供网络使用情况的统计数据。当发生可疑动作时，防火墙能进行适当的报警，并提供网络是否受到监测和攻击的详细信息。另外，收集一个网络的使用和误用情况也是非常重要的。首先的理由是可以清楚防火墙是否能够抵挡攻击者的探测和攻击，并且清楚防火墙的控制是否充足。而网络使用统计对网络需求分析和威胁分析等而言也是非常重要的。
- 防止内部信息的外泄。通过利用防火墙对内部网络的划分，可实现内部网重点网段的隔离，从而限制局部重点或敏感网络安全问题对全局网络造成的影响。再者，隐私是内部网络非常关心的问题，一个内部网络中不引人注意的细节可能包含了有关安全的线索而引起外部攻击者的兴趣，甚至因此而暴露了内部网络的某些安全漏洞。使用防火墙就可以隐蔽那些透露内部细节如Finger、DNS等服务。Finger显示了主机的所有用户的注册名、真名、最后登录时间和使用shell类型等。但是Finger显示的信息非常容

易被攻击者所获悉。攻击者可以知道一个系统使用的频繁程度，这个系统是否有用户正在连线上网，这个系统是否在被攻击时引起注意等。防火墙同样可以阻塞有关内部网络中的 DNS 信息，这样一台主机的域名和 IP 地址就不会被外界所了解。

除了安全作用，防火墙还支持具有 Internet 服务特性的企业内部网络技术体系 VPN（虚拟专用网）。

目前网络安全的三大主要问题是：以拒绝访问（DoS）为主要目标的网络攻击、以蠕虫（Worm）为主要代表的病毒传播和以垃圾电子邮件（SPAM）为代表的内容控制。这三大安全问题覆盖了网络安全方面的绝大部分问题。而这三大问题，传统的防火墙是无能为力的。新一代防火墙应该可加强放行数据的安全性，即既有包过滤的功能，又能在应用层进行代理。

### 2.3.2 防火墙的分类

防火墙发展到今天，分类的方式林林总总，下面详细介绍几种常用的分类方式。

#### 1．按防火墙的软、硬件形式分

按防火墙的软、硬件形式来分，可以分为软件防火墙和硬件防火墙。最初的防火墙与我们平时所看到的集线器、交换机一样，都属于硬件产品。其在外观上与平常所见到的集线器和交换机类似，只是接口比较少，分别用于连接内、外部网络，这是由防火墙的基本作用决定的。

随着防火墙应用的逐步普及和计算机软件技术的发展，为了满足不同层次用户对防火墙技术的需求，许多网络安全软件厂商开发出基于纯软件的防火墙，俗称“个人防火墙”。之所以说它是“个人防火墙”，那是因为其安装在主机中，只对一台主机进行防护，而不是针对整个网络。

#### 2．按防火墙技术分

按照防火墙的技术划分，又可以分为以下两类。

- 包过滤（Packet filtering）型。包过滤型防火墙工作在 OSI 网络参考模型的网络层和传输层，其根据数据包头源地址、目的地址、端口号和协议类型等标志确定是否允许通过。只有满足过滤条件的数据包才被转发到相应的目的地，其余的数据包则被从数据流中丢弃。
- 应用代理（Application Proxy）型。应用代理型防火墙工作在 OSI 的最高层，即应用层。其特点是完全“阻隔”了网络通信流，通过对每种应用服务编制专门的代理程序，以实现监视和控制应用层通信流的作用。

#### 3．按防火墙结构分

按照防火墙的结构，可以将其划分为以下几类。

- 单一主机防火墙。该防火墙是最传统的防火墙，其独立于其他网络设备，位于网络边界。这种防火墙其实与一台计算机的结构差不多，同样包括 CPU、内存、硬盘等基本组件。它与一般计算机最主要的区别是，一般防火墙都集成了两个以上的以太网卡，因为其需要连接一个以上的内、外部网络。其中的硬盘就是用来存储防火墙所用的基本程序，如包过滤程序和代理服务器程序等，有的防火墙还把日志记录也记录在此硬盘上。

- 路由器集成式防火墙。随着防火墙技术的发展及应用需求的提高，原来作为单一主机的防火墙现在已发生了许多变化。最明显的变化就是现在许多中、高档的路由器中已集成了防火墙功能，有的防火墙已不再是一个独立的硬件实体，而是由多个软、硬件组成的系统，这种防火墙，俗称“路由器集成式防火墙”。
- 分布式防火墙。分布式防火墙不再是位于网络边界，而是渗透于网络的每一台主机，对整个内部网络的主机实施保护。在网络服务器中，通常会安装一个用于防火墙的系统管理软件，在服务器及各主机上安装有集成网卡功能的 PCI 防火墙卡。这样一块防火墙卡就同时兼有网卡和防火墙的双重功能，即可彻底保护内部网络。各主机把任何其他主机发送的通信连接都视为“不可信”的，都须要严格过滤。而不是像传统边界防火墙那样，仅对外部网络发出的通信请求“不信任”。

#### 4．按防火墙的应用部署位置分

按照防火墙在具体应用中部署的位置，可以划分为以下几类。

- 边界防火墙。该类防火墙是最传统的防火墙，其位于内、外部网络的边界，它的作用是对内、外部网络实施隔离，保护边界内部网络。这类防火墙一般都是硬件类型的，价格较贵，性能较好。
- 个人防火墙。该类防火墙安装于单台主机中，也只是防护单台主机。这类防火墙应用于广大的个人用户，通常为软件防火墙，价格便宜，性能较差。
- 混合式防火墙。该类防火墙可以说就是“分布式防火墙”或者“嵌入式防火墙”，是一整套防火墙系统，由若干个软、硬件组件组成，分布于内、外部网络边界和内部各主机之间。它既对内、外部网络之间的通信进行过滤，又对网络内部各主机间的通信进行过滤；属于最新的防火墙技术之一，性能最好，价格也最贵。

#### 5．按防火墙性能分

如果按防火墙的性能来分，可以分为百兆级防火墙和千兆级防火墙两类。因为防火墙通常位于网络边界，所以不可能只是十兆级的，这主要是指防火墙的通道带宽，或者说是吞吐率。当然通道带宽越宽，性能越高，这样的防火墙因包过滤或应用代理所产生的延时也就越小，对整个网络通信性能的影响也就越小。

### 2.3.3 传统防火墙技术

传统的防火墙基数种类繁多，各有各的特点，因而在应用中也应区别对待，充分发挥其优势。下面分别介绍各种传统防火墙的技术及其特点。

#### 1．数据包过滤防火墙技术

数据包过滤（Packet Filtering）技术是在网络层对数据包进行选择，选择的依据是系统内设置的过滤逻辑，被称为访问控制表（Access Control Table）。通过检查数据流中每个数据包的源地址、目的地址、所用的端口号、协议状态等因素，或它们的组合来确定是否允许该数据包通过。数据包过滤防火墙的逻辑简单、价格便宜、易于安装和使用、网络性能和透明性好，其通常安装在路由器上。路由器是内部网络与 Internet 连接必不可少的设备，因此在原有网络上增加这样的防火墙几乎不需要任何额外的费用。

包过滤的优点是一个过滤路由器能协助保护整个网络，数据包过滤对用户透明，过滤路由器速度快、效率高；缺点是不能彻底防止地址欺骗，一些应用协议不适合于数据包过滤，正常的数据包过滤路由器无法执行某些安全策略。

2．应用层网关防火墙技术

应用层网关（Application Level Gateways）是在网络应用层上建立协议过滤和转发功能。其针对特定的网络应用服务协议使用指定的数据过滤逻辑，并在过滤的同时，对数据包进行必要的分析、登记和统计，形成报告。实际中的应用网关通常安装在专用工作站系统上。

数据包过滤和应用网关防火墙有一个共同的特点，就是仅仅依靠特定的逻辑判定是否允许数据包通过，一旦满足逻辑，则防火墙内外的计算机系统建立直接联系，防火墙外部的用户便有可能直接了解防火墙内部的网络结构和运行状态，这有利于实施非法访问和攻击。

3．代理防火墙技术

代理服务（Proxy Service）也称链路级网关或TCP通道（Circuit Level Gateways or TCP Tunnels），也有人将其归于应用层网关一类，是一种针对数据包过滤和应用网关技术存在的缺点而引入的防火墙技术。其特点是将所有跨越防火墙的网络通信链路分为两段。防火墙内外计算机系统间应用层的“连接”，由两个终止代理服务器上的“连接”来实现，外部计算机的网络链路只能到达代理服务器，从而起到了隔离防火墙内外计算机系统的作用。此外，代理服务也对过往的数据包进行分析、注册登记，形成报告，同时当发现被攻击迹象时会向网络管理员发出警报，并保留攻击痕迹。

### 2.3.4 新一代防火墙的技术特点

新一代防火墙的目的主要是综合包过滤和代理技术，克服二者在安全方面的缺陷；能从数据链路层一直到应用层施加全方位的控制；实现 TCP/IP 协议的微内核，从而在 TCP/IP 协议层进行各项安全控制；基于上述微内核，使速度超过传统的包过滤防火墙；提供透明代理模式，减轻客户端的配置工作；支持数据加密、解密（DES 和 RSA），提供对虚拟网 VPN 的强大支持；内部信息完全隐藏；产生一个新的防火墙理论。

新一代防火墙技术不仅覆盖了传统包过滤防火墙的全部功能，而且在全面对抗 IP 欺骗、SYN Flood、ICMP、ARP 等攻击手段方面有显著优势，增强了代理服务，并使其与包过滤相融合，再加上智能过滤技术，使防火墙的安全性提升到又一高度。下面将分别介绍几种常见的新一代防火墙技术。

1．新一代分布式防火墙技术

针对传统边界防火墙的缺陷，专家提出了分布式防火墙的概念。从狭义和与边界防火墙产品对应来讲，分布式防火墙产品是指那些驻留在网络中主机如服务器或桌面客户机并对主机系统自身提供安全防护的软件产品；从广义来讲，分布式防火墙是一种新的防火墙体系结构，其包含如下几种产品。

- 网络防火墙。用于内部网与外部网之间（即传统的边界防火墙）和内部网子网之间的防护产品，后者区别于前者的一个特征是须支持内部网可能有的 IP 和非 IP 协议。
- 主机防火墙。对于网络中的服务器和桌面机进行防护，这些主机的物理位置可能在内部网中，也可能在内部网外，如托管服务器或移动办公的便携机。

- 中心管理。边界防火墙只是网络中的单一设备，管理是局部的。对分布式防火墙来说，每个防火墙作为安全监测机制可以根据安全性的不同要求布置在网络中任何需要的位置上，但总体安全策略又是统一策划和管理的，安全策略的分发及日志的汇总都是中心管理应具备的功能。中心管理是分布式防火墙系统的核心和重要特征之一。

分布式防火墙克服了传统防火墙的缺陷，其优势在于：在网络内部增加了另一层安全，有效地抵御来自内部的攻击，消除网络边界上的通信瓶颈和单一故障点，支持基于加密和认证的网络应用；与拓扑无关，支持移动计算。

分布式防火墙主要应用在企业的网络和服务器主机，在于堵住内部网的漏洞，解决来自企业内部网的攻击。分布式防火墙实施在企业各个网络端点上，克服了传统防火墙的缺陷，有效保护了主机，适应了新的网络应用的需要。

#### 2．新一代嵌入式防火墙技术

嵌入式防火墙就是内嵌于路由器或交换机的防火墙。嵌入式防火墙是某些路由器的标准配置。用户可以购买防火墙模块，安装到已有的路由器或交换机中。嵌入式防火墙也被称为阻塞点防火墙。由于互联网使用的协议多种多样，所以不是所有的网络服务都能得到嵌入式防火墙的有效处理。嵌入式防火墙工作于 IP 层，所以无法保护网络免受病毒、蠕虫和特洛伊木马程序等来自应用层的威胁。就本质而言，嵌入式防火墙常常是无监控状态的，它在传递信息包时并不考虑以前的连接状态。

嵌入式防火墙弥补并改善各类安全能力不足的企业边缘防火墙、防病毒程序、基于主机的应用程序、入侵检测报警程序以及网络代理程序而设计，确保企业内部与外部的网络具有以下功能：不论企业局域网的拓扑结构如何变更，防护措施都能延伸到网络边缘，为网络提供保护；基于硬件、能够防范入侵的安全特性能独立于主机操作系统与其他安全性程序运行，甚至在安全性较差的宽带链路上也能实现安全移动与远程接入，可管理的执行方式使企业安全性能够被用户策略而非物理设施来进行定义。

一套嵌入式防火墙安全性解决方案能够为那些想在家里访问公司局域网的远程办公用户提供保护。帮助企业确保网络最薄弱和未保护领域的安全，如笔记本电脑和远程 PC 机，实行集中式管理的嵌入式客户机方案，并实现了跨越企业边缘防火墙的可靠网络连接，为企业和政府站点提供天衣无缝的安全性。

#### 3．新一代智能防火墙技术

智能防火墙从技术特征上，是利用统计、记忆、概率和决策的智能方法来对数据进行识别，并达到访问控制的目的。新的数学方法，消除了匹配检查所需要的海量计算，高效发现网络行为的特征值，直接进行访问控制。由于这些方法多是人工智能学科采用的方法，因此被称为智能防火墙。其关键技术主要如下。

- 防攻击技术。智能防火墙能智能识别恶意数据流量，并有效地阻断恶意数据攻击。智能防火墙可以有效地解决 SYN Flooding、Land Attack、UDP Flooding、Fraggle Attack、Ping Flooding、Smurf、Ping of Death、Unreachable Host 等攻击。防攻击技术还可以有效地切断恶意病毒或木马的流量攻击。
- 防扫描技术。智能防火墙能智能识别黑客的恶意扫描，并有效地阻断或欺骗恶意扫描者。对目前已知的如 ISS、SSS、NMAP 等扫描工具，智能防火墙可以防止被扫描。防

扫描技术还可以有效地解决恶意代码的恶意扫描攻击。

- 防欺骗技术。智能防火墙提供基于 MAC 的访问控制机制，可以防止 MAC 欺骗和 IP 欺骗，支持 MAC 过滤和 IP 过滤。将防火墙的访问控制扩展到 OSI 的第二层。
- 入侵防御技术。智能防火墙为了解决准许放行包的安全性，对准许放行的数据进行入侵检测，并提供入侵防御保护。入侵防御技术采用了多种检测技术，特征检测可以准确检测已知的攻击，特征库涵盖了目前流行的网络攻击；异常检测基于对监控网络的自学习能力，可以有效地检测新出现的攻击；检测引擎中还集成了针对缓冲区溢出等特定攻击的检测。智能防火墙完成了深层数据包监控，并能阻断应用层攻击。
- 包擦洗和协议正常化技术。智能防火墙支持包擦洗技术，对 IP、TCP、UDP、ICMP 等协议的擦洗，实现协议的正常化，消除潜在的协议风险和攻击。这些方法对消除 TCP/IP 协议的缺陷和应用协议的漏洞所带来的威胁，效果显著。
- 入侵防护。智能防火墙为了解决准许放行包的安全性，对准许放行的数据进行入侵检测，并提供入侵防御保护，这样就完成了深层数据包监控，并能阻断应用层攻击。

总之，与传统防火墙相比，智能防火墙在保护网络和站点免受黑客的攻击、阻断病毒的恶意传播、有效监控和管理内部局域网、保护必需的应用安全、提供强大的身份认证授权和审计管理等方面，都有广泛的应用价值。

### 2.3.5 防火墙技术的发展趋势

随着新的网络攻击的出现，防火墙技术也有一些新的发展趋势，这主要可以从包过滤技术、防火墙体系结构和防火墙系统管理三方面来体现。

#### 1. 防火墙包过滤技术

- 用户身份验证防火墙。一些防火墙厂商把在 AAA（Authentication、Authorization、Accounting，认证、授权、记账）系统上运用的用户认证及其服务扩展到防火墙中，使其拥有可以支持基于用户角色的安全策略功能。该功能在无线网络应用中非常必要。具有用户身份验证的防火墙通常是采用应用级网关技术的，包过滤技术的防火墙不具有。用户身份验证功能越强，它的安全级别越高，但它给网络通信带来的负面影响也越大，因为用户身份验证需要时间，特别是加密型的用户身份验证。
- 多级过滤技术。所谓多级过滤技术，是指防火墙采用多级过滤措施，并辅以鉴别手段。在分组过滤（网络层）一级，过滤掉所有的源路由分组和假冒的 IP 源地址；在传输层一级，遵循过滤规则，过滤掉所有禁止出或/和入的协议和有害数据包如 nuke 包、圣诞树包等；在应用网关（应用层）一级，能利用 FTP、SMTP 等各种网关，控制和监测 Internet 提供的所有通信服务。这是针对以上各种已有防火墙技术的不足而产生的一种综合型过滤技术，可以弥补以上各种单独过滤技术的不足。这种过滤技术在分层上非常清楚，每种过滤技术对应于不同的网络层，从这个概念出发，又有很多内容可以扩展，为将来的防火墙技术发展打下基础。
- 病毒防火墙。使防火墙具有病毒防护功能。现在通常被称之为病毒防火墙，当然目前主要还是在个人防火墙中体现，因其为纯软件形式，更容易实现。这种防火墙技术可以有效地防止病毒在网络中的传播，比等待攻击的发生更加积极。拥有病毒防护功能的防火墙可以大大减少企业的损失。

2．防火墙的体系结构

随着网络应用的增加，对网络带宽提出了更高的要求。这意味着防火墙要能够以非常高的速率处理数据。另外，在以后几年里，多媒体应用将会越来越普遍，要求数据穿过防火墙所带来的延迟要足够小。为了满足这种需要，一些防火墙制造商开发了基于 ASIC（Application Specific Integrated Circuit，特殊应用集成电路）的防火墙和基于网络处理器的防火墙。从执行速度的角度看来，基于网络处理器的防火墙也是基于软件的解决方案，需要在很大程度上依赖于软件的性能；但是由于这类防火墙中有一些专门用于处理数据层面任务的引擎，从而减轻了 CPU 的负担，该类防火墙的性能要比传统防火墙的性能好许多。

与基于 ASIC 的纯硬件防火墙相比，基于网络处理器的防火墙具有软件色彩，因而更加具有灵活性。基于 ASIC 的防火墙使用专门的硬件处理网络数据流，比起前两种类型的防火墙具有更好的性能。但是纯硬件的 ASIC 防火墙缺乏可编程性，这就使得其缺乏灵活性，从而跟不上防火墙功能的快速发展。理想的解决方案是增加 ASIC 芯片的可编程性，使其与软件更好地配合。这样的防火墙就可以同时满足来自灵活性和运行性能的要求。

3．防火墙的系统管理

总体来说，防火墙的系统管理有以下几种发展趋势。

- 集中式管理。集中式管理可以降低管理成本，并保证在大型网络中安全策略的一致性。快速响应和快速防御也要求采用集中式管理系统。
- 强大的审计功能和自动日志分析功能。这两点的应用可以更早地发现潜在的威胁并预防攻击的发生。日志功能还可让管理员有效地发现系统中存的安全漏洞，及时地调整安全策略等。不过具有这种功能的防火墙通常是比较高级的，早期的静态包过滤防火墙是不具有的。
- 网络安全产品的系统化。随着网络安全技术的发展，现在有一种体系结构的提法，叫做“建立以防火墙为核心的网络安全体系”。因为实践表明，仅使用现有的防火墙技术难以满足当前网络的安全需求。通过建立一个以防火墙为核心的安全体系，就可以为内部网络系统部署多道安全防线，各种安全技术各司其职，从各方面防御外来入侵。

### 2.3.6 防火墙的配置方式

防火墙的配置有三种：Dual-homed 方式、Screened- host 方式和 Screened-subnet 方式。

- Dual-homed 方式最简单。Dual-homed Gateway 放置在两个网络之间，这个 Dual-homed Gateway 又称为 bastion host。这种结构成本低，但是它有单点失败的问题。这种结构没有增加网络安全的自我防卫能力，而它往往是受黑客攻击的首选目标，它自己一旦被攻破，整个网络也就暴露了。
- Screened-host 方式中的 Screening router 为保护 Bastion host 的安全建立了一道屏障。它将所有进入的信息先送往 Bastion host，并且只接收来自 Bastion host 的数据作为出去的数据。这种结构依赖 Screening router 和 Bastion host，只要有一个失败，整个网络就暴露了。
- Screened-subnet 包含两个 Screening router 和两个 Bastion host。在公共网络和私有网络之间构成了一个隔离网，称之为“停火区”（DMZ，即 Demilitarized Zone），Bastion host

放置在“停火区”内。这种结构安全性好，只有当两个安全单元被破坏后，网络才被暴露，但是成本也很昂贵。

### 2.3.7 防火墙的实际安全部署建议

防火墙在实际的部署应用过程当中，经常部署在网关的位置，也就是经常部署在网内和网外的一个“中间分隔点”上，而就是在这样一个部署的环境中，仍存在着多种方式，且存在着许多“陷阱”，下面在该解决方案中明确给出了分析。

#### 1. 错误的防火墙部署方式

传统的防火墙部署方式可能所有人都认为非常简单，将防火墙部署于外部网络和内部网络之间。这个思路如果在内部网络中存在共享资源（比如说 FTP 服务器和 Web 服务器）的话，那么这将是一个非常危险的部署方式，如图 2-5 所示。理由其实非常简单，一旦这些共享服务器被黑客攻击和安装木马渗透病毒的话，那么内部网络的客户端及其资源将没有任何安全可言。因为在这种情况下，木马和病毒已经在内网中存在，而客户端和共享资源服务器在同一个网段，这无异于内网的安全隐患，防火墙对此无能为力，从而也失去了部署的意义。

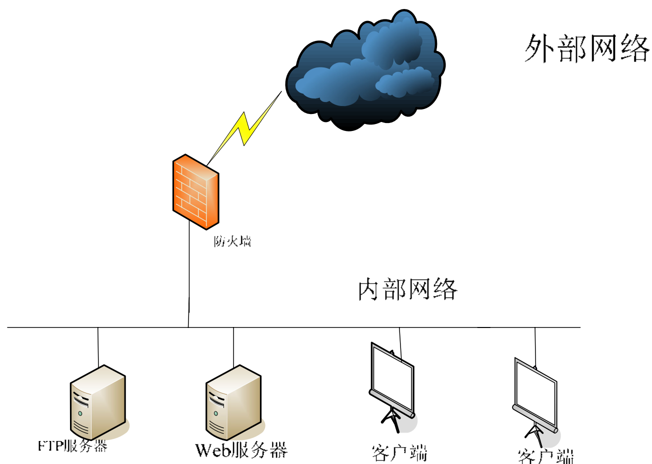



图 2-5 错误的防火墙部署方式

#### 2. 使用 DMZ

目前一个比较流行和正确的做法就是采用 DMZ 的防火墙部署方式，如图 2-6 所示。也就是在防火墙上多加一块网卡，将提供对外服务的服务器和内网的客户端严格地隔离开来，这样，即使有安全风险和漏洞在 DMZ 中出现，由此对内部网络造成的危害也可以得到很好的控制，从而避免了方案一的缺点。

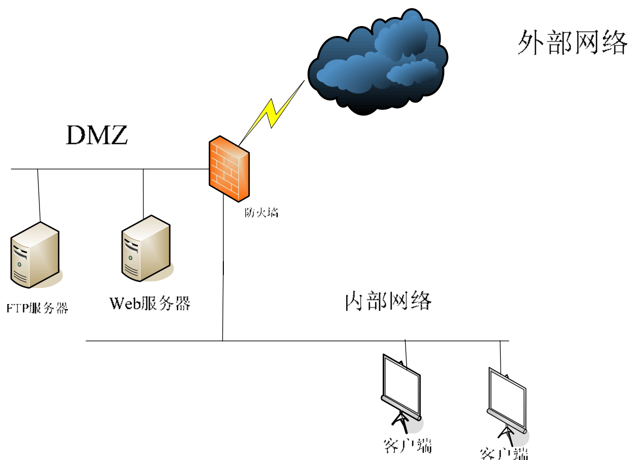



图 2-6 使用 DMZ 的防火墙部署方式

### 3. 使用 DMZ+二路防火墙

为了加强方案二中防火墙的安全强度，目前有些企业将图 2-6 的架构优化成图 2-7 的架构，也就是使用 DMZ+二路防火墙。另外，在此结构中选用防火墙，应尽量采用两家不同公司的产品，这样才有利于发挥这种架构的优势。

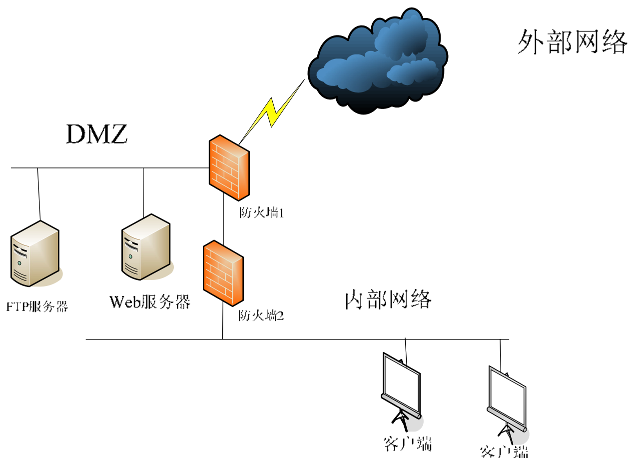



图 2-7　使用 DMZ+二路防火墙的部署方式

### 4. 通透式防火墙

在前面的几种方案中，防火墙本身就是一个路由器，在使用的过程中用户必须慎重地考虑到路由的问题。如果网络环境非常复杂或者是需要进行调整，则相应的路由需要进行变更，维护和操作起来有一定的难度和工作量。

通透式防火墙则可以比较好地解决上述问题，如图 2-8 所示。该类防火墙是一个桥接设备，并且在桥接设备上赋予了过滤的能力。由于桥接设备工作在 OSI 模型的第二层（也就是数据链路层），所以不会有任何路由的问题。并且，防火墙本身也不需要指定 IP 地址，因此，这种防火墙的部署能力和隐蔽能力都相当强，从而可以很好地应对黑客对防火墙自身的攻击，因为黑客很难获得可以访问的 IP 地址。

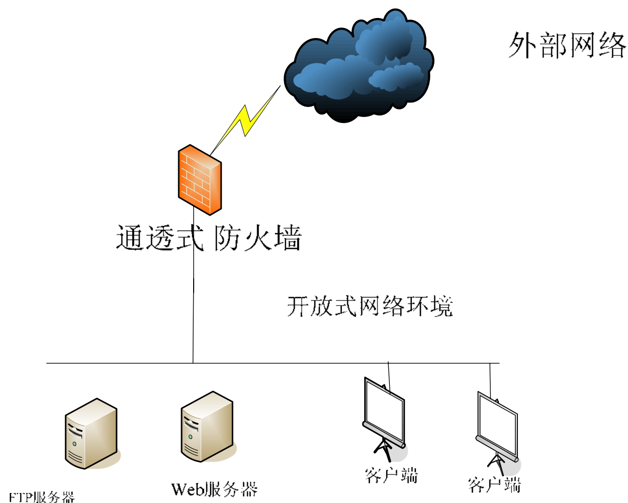



图 2-8　通透式防火墙部署方式

## 2.4 应用层防护：IDS/IPS

### 2.4.1 入侵检测系统简介

IDS（Intrusion Detection System，入侵检测系统）顾名思义便是对入侵行为的发觉，其通过对计算机网络或计算机系统中的若干关键点收集信息并对其进行分析，从中发现网络或系统中是否有违反安全策略的行为和被攻击的迹象。通常来说，其具有以下几个功能。

- 监控、分析用户和系统的活动。
- 核查系统配置和漏洞。
- 评估关键系统和数据文件的完整性。
- 识别攻击的活动模式并向网管人员报警。
- 对异常活动的统计分析。
- 操作系统审计跟踪管理，识别违反政策的用户活动。

按照技术以及功能来划分，入侵检测系统可以分为以下几类。

- 基于主机的入侵检测系统：其输入数据来源于系统的审计日志，一般只能检测该主机上发生的入侵。
- 基于网络的入侵检测系统：其输入数据来源于网络的信息流，能够检测该网段上发生的网络入侵。
- 采用上述两种数据来源的分布式入侵检测系统：能够同时分析来自主机系统审计日志和网络数据流的入侵检测系统，一般为分布式结构，由多个部件组成。

入侵检测系统的结构如图 2-9 所示。

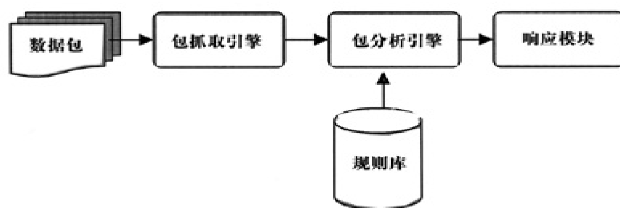



图 2-9　IDS 系统结构图

- 包抓取引擎：从网络上抓取数据包。
- 包分析引擎：对数据包做简单处理，如 IP 重组、TCP 流重组，并根据规则库判断是否为可疑或入侵的数据包。
- 规则库：是入侵检测系统的知识库，定义了各种入侵的知识。
- 响应模块：是当系统发现一个可疑的数据包时所采取的响应手段。

其中，包分析引擎是整个系统的核心所在，对入侵特征的检测在这里完成。

### 2.4.2　入侵检测技术的发展

自 1980 年产生 IDS 概念以来，已经出现了基于主机和基于网络的入侵检测系统，出现了基于知识的模型识别、异常识别和协议分析等入侵检测技术，并能够对百兆、千兆甚至更高流量的网络系统执行入侵检测。

入侵检测技术的发展已经历了以下四个主要阶段。

第一阶段是以基于协议解码和模式匹配为主的技术，其优点是对于已知的攻击行为非常有效，各种已知的攻击行为可以对号入座，误报率低；缺点是高超的安全采用变形手法或者新技术可以轻易躲避检测，漏报率高。

第二阶段是以基于模式匹配+简单协议分析+异常统计为主的技术，其优点是能够分析处理一部分协议，可以进行重组；缺点是匹配效率较低，管理功能较弱。这种检测技术实际上是在第

一阶段技术的基础上增加了部分对异常行为分析的功能。

第三阶段是以基于完全协议分析+模式匹配+异常统计为主的技术，其优点是误报率、漏报率和滥报率较低，效率高，可管理性强，并在此基础上实现了多级分布式的检测管理；缺点是可视化程度不够，防范及管理功能较弱。

第四阶段是以基于安全管理+协议分析+模式匹配+异常统计为主的技术，其优点是入侵管理和多项技术协同工作，建立全局的主动保障体系，具有良好的可视化、可控性和可管理性。以该技术为核心，可构造一个积极的动态防御体系，即IMS（Intrusion Managernent System，入侵管理系统）。新一代的入侵检测系统应该是具有集成HIDS和NIDS的优点、部署方便、应用灵活、功能强大，并提供攻击签名、检测、报告和事件关联等配套服务功能的智能化系统。

随着交换技术、加密信道技术和入侵技术的不断发展，对入侵检测技术的要求也越来越高，检测的方法手段也越来越复杂。现代入侵技术具有以下一些特点。

- 综合化和复杂化。
- 间接化。
- 规模化。
- 分布式。
- 范围广。

然而，目前 IDS 的检测模型始终落后于攻击者的新知识和技术手段，其主要表现在以下几个方面。

- 利用加密技术欺骗IDS。
- 躲避IDS的安全策略。
- 快速发动进攻，使IDS无法反应。
- 发动大规模攻击，使IDS判断出错。
- 直接破坏IDS。
- 智能攻击技术，边攻击边学习，变IDS为攻击者的工具。

因此，面对如此严峻的形势，入侵检测技术发展趋势如下。

- 分布式入侵检测，扩大检测范围和类别。
- 智能化入侵检测，自学习、自适应。
- 应用层入侵检测。
- 高速入侵检测。
- 标准化和系统化入侵检测。

目前，IDS发展的新趋势主要表现在两个方向上，一个是趋向构建入侵防御系统（Intrusion Prevention System，IPS）。IPS是在IDS中增加主动响应功能实现的，并以串联方式接入网络（IDS是以并联方式接入网络的），一旦发现有攻击行为，则立即响应，主动切断与攻击者的连接。IPS不仅具有入侵检测功能，还具有安全防护功能；二是趋向构建入侵管理系统（IMS）。IMS是IDS发展的另一个方向，IMS的目标是将入侵检测、脆弱性分析，以及入侵防御等多种功能集成到一个平台上进行统一管理。IMS技术是一个管理过程，在未发生攻击时，IMS主要考虑网络中的漏洞信息，评估和判断可能形成的攻击和将面临的威胁；在发生攻击或即将发生攻击时，不仅要检测出入侵行为，还要主动响应和防御入侵行为；在受到攻击后，还要深入分析入侵行为，并通过关联分析来判断可能出现的下一个攻击行为。

总而言之，入侵检测是一门综合性技术，既包括实时检测技术，也有事后分析技术。尽管用户希望通过部署 IDS 来增强网络安全，但不同的用户其需求也不同。由于攻击的不确定性，单一的 IDS 产品可能无法做到面面俱到。因此，IDS 的未来发展必然是多元化的，只有通过不断改进和完善才能更好地协助网络进行安全防御。

## 2.4.3 入侵检测技术的分类

从技术上讲，入侵检测技术大致分为基于知识的模式识别、基于知识的异常识别和协议分析三类。而主要的入侵检测方法有特征检测法、概率统计分析法和专家知识库系统。

#### 1．基于知识的模式识别

这种技术是通过事先定义好的模式数据库实现的，其基本思想是：首先把各种可能的入侵活动用某种模式表示出来，并建立模式数据库，然后监视主体的一举一动，当检测到主体活动违反事先定义的模式规则时，根据模式匹配原则判别是否发生了攻击行为。模式识别的关键是建立入侵模式的表示形式，同时，要能够区分入侵行为和正常行为。这种检测技术仅限于检测出已建立模式的入侵行为，属已知类型，对新类型的入侵是无能为力的，仍需改进。

#### 2．基于知识的异常识别

这种技术是通过事先建立正常行为档案库实现的，其基本思想是：首先把主体的各种正常活动用某种形式描述出来，并建立“正常活动档案”，当某种活动与所描述的正常活动存在差异时，就认为是“入侵”行为，进而被检测识别。异常识别的关键是描述正常活动和构建正常活动档案库。利用行为进行识别时，存在四种可能：一是入侵且行为正常；二是入侵且行为异常；三是非入侵且行为正常；四是非入侵且行为异常。根据异常识别思想，将第二种和第四种情况判定为“入侵”行为。这种检测技术可以检测出未知行为，并具有简单的学习功能。

以下是几种基于知识的异常识别的检测方法。

1）基于审计的攻击检测技术

这种检测方法是通过对审计信息的综合分析实现的，其基本思想是：根据用户的历史行为、先前的证据或模型，使用统计分析方法对用户当前的行为进行检测和判别，当发现可疑行为时，保持跟踪并监视其行为，同时向系统安全员提交安全审计报告。

2）基于神经网络的攻击检测技术

由于用户的行为十分复杂，要准确匹配一个用户的历史行为和当前的行为是相当困难的，这也是基于审计攻击检测的主要弱点。而基于神经网络的攻击检测技术则是一个对基于传统统计技术的攻击检测方法的改进方向，它能够解决传统的统计分析技术所面临的若干问题，例如，建立确切的统计分布、实现方法的普遍性、降低算法实现的成本和系统优化等问题。

3）基于专家系统的攻击检测技术

所谓专家系统，就是一个依据专家经验定义的推理系统。这种检测是建立在专家经验基础上的，它根据专家经验进行推理、判断，然后得出结论。例如，当用户连续三次登录失败时，可以把该用户的第四次登录视为攻击行为。

4）基于模型推理的攻击检测技术

攻击者在入侵一个系统时往往采用一定的行为程序，如猜测口令的程序。这种行为程序构成

了某种具有一定行为特征的模型，根据这种模型所代表的攻击意图的行为特征，可以实时地检测出恶意的攻击企图（尽管攻击者不一定都是恶意的）。用基于模型的推理方法，人们能够为某些行为建立特定的模型，从而监视具有特定行为特征的某些活动。根据假设的攻击脚本，这种系统就能检测出非法的用户行为。通常为了准确判断，要为不同的入侵者和不同的系统建立特定的攻击脚本。使用基于知识的模式识别和基于知识的异常识别所得出的结论差异较大，甚至得出相反的结论。这是因为，基于知识的模式识别的核心是维护一个入侵模式库，它对已知攻击可以详细、准确地报告出攻击类型，但对未知攻击却无能为力，而且入侵模式库必须不断更新；而基于知识的异常识别则是通过对入侵活动的检测得出结论的，它虽无法准确判断出攻击的手段，但可以发现更广泛的，甚至未知的攻击行为。

3．协议分析

这种检测方法是根据针对协议的攻击行为实现的，其基本思想是：首先把各种可能针对协议的攻击行为描述出来，然后建立用于分析的规则库，最后利用传感器检查协议中的有效荷载，并详细解析，从而实现入侵检测。这种检测技术能检测出更为广泛的攻击，包括已知的和未知的攻击行为。

入侵检测系统之间的区别体现在很多方面，按照信息源可以分为基于主机和基于网络的 IDS，按照控制策略可以分为集中式和分布式 IDS，按照响应方式可以分为主动响应和被动响应 IDS，按照处理的时机可以分为实时处理型和事后处理型 IDS，按照分析方法可分为误用检测型和异常检测型 IDS，我们在此主要介绍最为普遍的按照信息源和分析方法划分的两类入侵检测系统。

### 2.4.4 入侵检测系统的分类

1．基于主机的 IDS 和基于网络的 IDS

在入侵检测系统发展的早期阶段，网络远远没有今天这样流行，因此早期的入侵检测系统大多是基于主机的系统。基于主机的入侵检测系统通常应用两种类型的信息源：操作系统审计踪迹和系统日志。操作系统审计踪迹由操作系统内核产生，这些审计踪迹是系统活动信息的集合，是对系统事件的真实记录。由于操作系统本身提供了对审计踪迹的保护机制，因此作为入侵检测的信息源，操作系统审计踪迹的可靠性能得到很好的保证，但审计数据过于庞杂并且不易理解是其弱点所在。系统日志是一个反映各种各样的系统事件和设置的文件，由于日志文件通常是由应用程序产生，而且通常存储在不受保护的目录里，与操作系统审计踪迹相比，安全性不够好，但是系统日志结构简单（比如作为文本文件形式存在），理解起来相对容易，而安全性问题可以通过日志文件重定向等方法来解决，因此日志文件仍然是基于主机的入侵检测系统最常用的信息源之一，对日志文件在入侵检测系统中应用的研究也是当前的研究热点之一。基于主机的入侵检测系统的优点包括：对入侵事件的观察更为细腻，理解更为准确，可以观察到入侵事件的后果，可以检测到网络入侵检测系统检测不到的入侵；不受网络信息流加密和交换网络的影响；可以检测到特洛伊木马等破坏软件完整性的入侵。所存在的不足主要有：占用所监视主机的系统资源，影响系统运行效率；无法检测针对网络发起的协同入侵；本身容易受到入侵而失效。

随着网络的飞速发展，基于网络的入侵检测系统开始走向前台，成为入侵检测研究的热点和主流，目前的入侵检测系统大多是基于网络的入侵检测系统。基于网络的入侵检测系统信息源来自网络，系统通过对网络数据流进行捕获、分析，以判断是否受到入侵。在体系结构上，基于网络的

入侵检测系统通常包含一系列传感器（sensor）和中央控制台，这些传感器负责监视网络数据流，做局部的分析和判断，并向中央控制台报告。这些传感器通常被设计成隐藏模式运行，因此安全性较好。基于网络的入侵检测系统的主要优点包括：作用范围较广。与仅能监视单一主机的主机型入侵检测系统不同，基于网络的入侵检测系统可以部署在网段的关键位置，监控流经该网段所有主机的网络通信流，保护该网段的所有主机，这对局域网用户特别实用；本身的抗入侵性能较好，自身安全性较高；提供实时的网路监视，对入侵反应迅速；对现有网络影响很小；操作系统无关性。所存在的主要不足之处在于：高速网络环境下的数据报丢失问题；交换网络环境下以及 VPN 环境下数据报文的加密问题，随着越来越多的企业组织使用 VPN（Virtual Private Network，虚拟专用网），这个问题将会变得更加突出；检测精确度较差，容易被高明的黑客欺骗。

基于主机和基于网络的入侵检测系统都有各自的优势，两者相互补充。这两种方式都能发现对方无法检测到的一些入侵行为。从某个重要服务器的键盘发动的本地入侵并不经过网络，因此就无法通过基于网络的入侵检测系统检测到，只能通过使用基于主机的入侵检测系统来检测。基于网络的入侵检测系统通过检查所有的数据包的包头（header）来进行检测，而基于主机的入侵检测系统并不查看报文头部值。许多基于 IP 的拒绝服务入侵和碎片入侵，只能通过查看它们通过网络传输时的包首标才能识别。基于网络的入侵检测系统可以研究负载的内容，查找特定入侵中使用的命令或语法，这类入侵可以被实时检查包序列的入侵检测系统迅速识别。而基于主机的系统无法看到负载，因此也无法识别嵌入式的负载入侵。综合使用基于主机和基于网络这两种方式能够发挥更好的检测性能。比如基于主机的入侵检测系统使用系统日志作为检测依据，因此它们在确定入侵是否已经取得成功时与基于网络的检测系统相比具有更大的准确性。在这方面，基于主机的入侵检测系统对基于网络的入侵检测系统是一个很好的补充，人们完全可以使用基于网络的入侵检测系统提供早期报警，而使用基于主机的入侵检测系统来验证入侵是否取得成功。

从上面的分析可以看出，基于主机的入侵检测系统和基于网络的入侵检测系统具有较强的互补性。将基于主机的入侵检测系统和基于网络的入侵检测系统结合起来，取长补短，优势互补，对于提高入侵检测系统的检测性能很有帮助。

#### 2．基于知识检测的 IDS 和基于行为检测的 IDS

基于知识的检测也称为误用检测，误用检测对检测的系统活动进行分析，试图发现那些与预定义好的入侵特征相匹配的事件和事件集。与入侵相对应的模式被称为特征，所以误用检测也被称为基于特征的检测。在目前的误用检测产品中，最常见的实现形式是将每一个入侵事件行为定义为一个特征描述，所有入侵行为特征描述就组成一个特征库。误用检测系统通过对事件进行分析，提取出事件的特征模式，并与特征库进行匹配，由匹配结果来判断是否有入侵行为发生。规则匹配是最常见的误用检测手段，目前也有比较复杂的误用检测技术，包括基于状态的分析技术和状态转换技术，都在一定程度上对简单的规则匹配技术进行了改进和提高。误用检测技术的优点有：能准确迅速地检测到入侵模式库里已经定义好的入侵模式，而不会产生太多的误报；检测结果清晰明朗，容易对入侵事件有准确的定位；类似于杀毒软件的工作模式，便于升级维护。存在的主要不足之处包括：对入侵模式库的依赖性很强，只能检测到已知的入侵模式，对新型入侵无能为力，容易产生漏报；入侵模式库的完备性难以保证，存在难以管理以及搜索的效率问题。

基于行为的检测也称为异常检测，异常检测技术基于用户行为和进程行为都有较大程度的稳定性这样一个前提。如果建立了用户行为和进程行为的正常模式，当出现较大的偏差异常时，则可以认为有恶意的入侵行为发生。异常检测的关键问题是正常模式的建立以及对偏差的检测和判

定，在实现技术上，异常检测系统首先收集一段时间正常操作活动的历史数据，建立起代表用户、主机、网络连接的正常行为轮廓，然后收集事件数据并使用不同的方法来判断所检测的事件活动是否偏离了正常行为轮廓。异常检测技术是目前入侵检测技术研究的热点领域，其所能达到的对未知入侵行为的检测能力令人神往，虽然目前还不能准确地做到这一点，但有一些研究至少可以表明这一点是能够做到的。在异常检测领域主要的方法有：统计方法、机器学习方法、数据挖掘方法、聚类方法等。异常检测技术的优点有：不依赖于具体的入侵知识，而是基于行为，可以检测到新型的入侵方式；不存在入侵模式库，避免了烦琐费时的搜索匹配工作，效率较高。存在的主要不足之处：系统、用户、网络行为的复杂性导致正常模式的建立较为困难；统计方法的应用要有一定的前提；异常行为的判断很难做到准确，误报率较高。

误用检测和异常检测代表了两种不同的研究思路，误用检测的核心问题是对入侵行为进行模式提取，并试图建立一个尽可能包含更多入侵行为模式的特征库，之后对所捕获到的事件进行分析匹配，以发现与入侵模式库里模式相匹配的入侵行为；异常检测的核心问题是正常模式的建立和异常判定标准的建立，先建立起正常的模式，之后对所捕获的事件进行分析判定，看其是否已经达到异常的标准。这两种思路虽然不同，但这两种技术可以有效地结合起来，优势互补，以实现对入侵事件更准确的检测，比如可以把异常检测的输出作为误用检测特征库的更新输入来源，而误用检测对入侵行为的判定也可以作为异常检测对异常行为判定的一个参考。目前商业化的入侵检测产品大都是将误用检测技术和异常检测技术结合起来，以误用检测模块作为主体，异常检测模块作为有益的补充。

综上所述，不论是何种入侵检测系统都存在这样或那样的不足，因而需要不断改进和完善，一个较为理想的入侵检测系统应具备以下特征。

- 准确性。检测系统对发现的攻击行为不应出现误报和漏报现象。
- 可靠性。一个检测系统对管理员应该是透明的，并且能在无人监控的情况下正确运行，只有这样才可以运行在被检测的系统环境中。
- 容错性。检测系统必须具有良好的容错性，不论所监控的系统处于何种状态，检测系统本身必须具备完整性，保证检测用的知识库系统不会受到干扰和破坏。
- 可用性。检测系统的整体性能不应受系统状态的变化而产生较大波动或严重降低。
- 可验证性。检测系统必须允许管理员适时监视攻击行为。
- 安全性。检测系统能保护自身安全和具有较强的抗欺骗攻击的能力。
- 可适应性。检测系统可随时跟踪系统环境的变化和及时调整检测策略。
- 灵活性。检测系统可根据具体情况，定制不同的且与防御机制相适应的使用模式。

### 2.4.5 入侵防御系统（IPS）

随着网络入侵事件的不断增加和黑客攻击水平的不断提高，一方面企业网络感染病毒、遭受攻击的速度日益加快，另一方面企业网络受到攻击做出响应的时间却越来越滞后。解决这一矛盾，传统的防火墙或入侵检测技术（IDS）显得力不从心，这就需要引入一种全新的技术——入侵防御（Intrusion Prevention System，IPS）。

入侵防御系统是网络安全设施，是对防病毒软件（Antivirus Programs）和防火墙（Packet Filter，Application Gateway）的补充。入侵防御系统是一部能够监视网络或网络设备的网络资料传输行为的计算机网络安全设备，能够即时地中断、调整或隔离一些不正常或是具有伤害性的网络资料

传输行为。

### 2.4.6 IPS 的发展

IPS 是这段时间网络安全业内比较热门的一个词，这种既能及时发现又能实时阻断各种入侵行为的安全产品，自面世那天起，就受到各大安全厂商和用户的广泛关注。入侵防御系统（IPS）就是入侵检测系统（IDS）的升级产品，有了 IPS，就可以替代以前的 IDS 系统，这也正是 Gartner 在 2003 年发表那篇著名的“IDSisdead”的理由。

从入侵防御系统的起源来看，这个“升级说”似乎有些道理：NetworkICE 公司在 2000 年首次提出了 IPS 这个概念，并于 2000 年的 9 月 18 日推出了 BlackICEGuard，这是一个串行部署的 IDS，直接分析网络数据并实时对恶意数据进行丢弃处理。但这种概念一直受到质疑，自 2002 年 IPS 概念传入国内起，IPS 这个新型的产品形态就不断地受到挑战，而且各大安全厂商、客户都没有表现出对 IPS 的兴趣，普遍的一个观点是：在 IDS 基础上发展起来的 IPS 产品，在没能解决 IDS 固有问题的前提下，是无法得到推广应用的。

这个固有问题就是“误报”和“滥报”，IDS 的用户常常会有这种苦恼：IDS 界面上充斥着大量的报警信息，经过安全专家分析后，被告知这是误警。但在 IDS 旁路检测的部署形式下，这些误警对正常业务不会造成影响，仅需要花费资源去做人工分析。而串行部署的 IPS 就完全不一样了，一旦出现了误报或滥报，触发了主动的阻断响应，用户的正常业务就有可能受到影响，这是所有用户都不愿意看到和接受的。正是这个原因，导致了 IPS 概念在 2005 年之前的国内市场表现平淡。随着时间的推进，自 2006 年起，大量的国外厂商的 IPS 产品进入国内市场，各本土厂商和用户都开始重新关注起 IPS 这一并不新鲜的“新”概念。

明确了 IPS 的主线功能是深层防御、精确阻断后，IPS 未来发展趋势也就明朗化了：不断丰富和完善 IPS 可以精确阻断的攻击种类和类型，并在此基础之上提升 IPS 产品的设备处理性能。

在提升性能方面存在的一个悖论就是：需提升性能，除了在软件处理方式上优化外，硬件架构的设计也是一个非常重要的方面，目前的 ASIC/NP 等高性能硬件，都是采用嵌入式指令+专用语言开发，将已知攻击行为的特征固化在电子固件上，虽然能提升匹配的效率，但在攻击识别的灵活度上过于死板（对变种较难发现），在新攻击特征的更新上有所滞后（需做特征的编码化）。而基于开放硬件平台的 IPS 由于采用的是高级编程语言，不存在变种攻击识别和特征更新方面的问题，但在性能上存在处理效率瓶颈：暂时达不到电信级骨干网络的流量。所以，入侵防御系统的未来发展方向应该有以下两个方面。

第一，更加广泛的精确阻断范围：扩大可以精确阻断的事件类型，尤其是针对变种以及无法通过特征来定义的攻击行为的防御。

第二，适应各种组网模式：在确保精确阻断的情况下，适应电信级骨干网络的防御需求。

### 2.4.7 IPS 的技术特征

IPS 具有以下技术特征。

- 嵌入式运行：只有以嵌入模式运行的 IPS 设备才能够实现实时的安全防护，实时阻拦所有可疑的数据包，并对该数据流的剩余部分进行拦截。
- 深入分析和控制：IPS 必须具有深入分析能力，以确定哪些恶意流量已经被拦截，根据

攻击类型、策略等来确定哪些流量应该被拦截。

- 入侵特征库：高质量的入侵特征库是 IPS 高效运行的必要条件，IPS 还应该定期升级入侵特征库，并快速应用到所有传感器。
- 高效处理能力：IPS 必须具有高效处理数据包的能力，对整个网络性能的影响保持在最低水平。

### 2.4.8 IPS 的功能特点

入侵防御系统作为串接部署的设备，确保用户业务不受影响是一个重点，错误的阻断必定意味着影响正常业务，在错误阻断的情况下，各种所谓扩展功能、高性能都是一句空话。这就引出了 IPS 设备应该关心的重点——精确阻断，即精确判断各种深层的攻击行为，并实现实时的阻断。

精确阻断解决了自 IPS 概念出现以来用户和厂商的最大困惑：如何确保 IPS 无误报和滥报，使得串接设备不会形成新的网络故障点。而作为一款防御入侵攻击的设备，毫无疑问，防御各种深层入侵行为是第二个重点，这也是 IPS 系统区别于其他安全产品的本质特点；这也给精确阻断加上了一个修饰语：保障深层防御情况下的精确阻断，即在确保精确阻断的基础上，尽量多地发现攻击行为（如 SQL 注入攻击、缓冲区溢出攻击、恶意代码攻击、后门、木马、间谍软件），这才是 IPS 发展的主线功能。

常用的攻击检测方法有两种，一种方法是通过定义攻击行为的数据特征来实现对已知攻击的检测，其优势是技术上实现简单、易于扩充、可迅速实现对特定新攻击的检测和拦截；但仅能识别已知攻击、抗变种能力弱。另一种方法是通过分析攻击产生原理，定义攻击类型的统一特征，能准确识别基于相同原理的各种攻击、不受攻击变种的影响，但技术门槛高、扩充复杂、应对新攻击速度有限。

融合“基于特征的检测机制”和“基于原理的检测机制”形成的“柔性检测”机制，其最大特点就是基于原理的检测方法与基于特征的检测方法并存，有机组合了两种检测方法的优势。这种融合不仅是一个两种检测方法的大融合，而且细分到对攻击检测防御的每一个过程中，在抗躲避的处理、协议分析、攻击识别等过程中都包含了动态与静态检测的融合。

通过运用柔性检测机制，入侵防御系统进一步增强了设备的抗躲避能力、精确阻断能力、变形攻击识别能力和对新攻击应变能力，提高了精确检测的覆盖面。

扩展功能和高性能，也是入侵防御系统所必须关注的内容，但也要符合产品的主线功能发展趋势。如针对 P2P 的限制：P2P 作为一种新兴的下载手段，得到了极为广泛的运用，但由无限制的 P2P 应用会影响网络的带宽消耗，并且还随之带来知识产权、病毒等多种相关问题。而实现对 P2P 的控制和限制，需要较为深入的应用层分析，交给 IPS 来限制、防范，是一个比较恰当的选择。而 ACL 控制、路由、NAT 等，这些都是防火墙可以完成的工作，在 IPS 上来实现这些功能，就有画蛇添足之嫌了。

性能表现是 IPS 的又一重要指标，但这里的性能应该是更广泛含义上的性能：包括了最大的参数表现和异常状况下的稳定保障。也就是说，性能除了需要关注诸如“吞吐率多大？”，“转发时延多长？”，“一定背景流下检测率如何？”等性能参数表现外，还需要关注：“如果出现了意外情况，怎样/多快能恢复网络的正常通信？”，这个问题也是 IPS 出现之初被质疑的一个重点。

实时检测与主动防御是 IPS 最为核心的设计理念，也是其区别于防火墙和 IDS 的立足之本。

为实现这一理念，IPS 在以下四个方面实现了技术突破，形成了不可低估的优势。

- 在线安装（In-Line）。IPS 保留 IDS 实时检测的技术与功能，但是却采用了防火墙式的在线安装，即直接嵌入到网络流量中，通过一个网络端口接收来自外部系统的流量，经过检查确认其中不包含异常活动或可疑内容后，再通过另外一个端口将它传送到内部系统中。
- 实时阻断（Real-time Interdiction）。IPS 具有强有力的实时阻断功能，能够预先对入侵活动和攻击性网络流量进行拦截，避免其造成任何损失。
- 先进的检测技术（Advanced Detection Technology）。主要是并行处理检测和协议重组分析。所谓并行处理检测是指所有流经 IPS 的数据包，都采用并行处理方式进行过滤器匹配，实现在一个时钟周期内，遍历所有数据包过滤器；而协议重组分析是指所有流经 IPS 的数据包，必须首先经过硬件级预处理，完成数据包的重组，确定其具体应用协议。然后，根据不同应用协议的特征与攻击方式，IPS 对于重组后的包进行筛选，将可疑者送入专门的特征库进行比对，从而提高检测的质量和效率。
- 特殊规则植入功能（Build-in Special Rule）。IPS 允许植入特殊规则以阻止恶意代码。IPS 能够辅助实施可接收应用策略（AUP），如禁止使用对等的文件共享应用和占有大量带宽的免费互联网电话服务工具等。
- 自学习与自适应能力（Self-study & Self-adaptation Ability）。为了应对黑客们处心积虑、花样翻新的攻击手段，IPS 必须具有人工智能的自学习与自适应能力。能够根据所在网络的通信环境和被入侵状况，分析和抽取新的攻击特征以更新特征库，自动总结经验，定制新的安全防御策略。

客观地说，IPS 也存在如下缺点。

- 总体拥有成本（TOC）高。浩大的高可用性（HA）实时计算需求决定了 IPS 必须选用高端的专用计算设备，但是可观的总体拥有成本却使不少用户望而却步。
- 单点故障（Single-point Fault）。IPS 的阻断能力决定其必须采用网络嵌入模式，而这就可能造成单点故障。
- 性能瓶颈（Performance Bottle-neck）。即使 IPS 设备不出现故障，它仍然是一个潜在的网络瓶颈，不仅会增加滞后时间，而且会降低网络的效率，因此，绝大多数高端 IPS 产品供应商都通过使用自定义硬件（FPGA、网络处理器或者 ASIC 芯片）来提高 IPS 的运行效率，以减少其对于业务网络的负面影响。
- 误报（False positive）与漏报（False negatives）后果同样严重。在网络流量几乎成几何级数增加的情况下，一旦生成警报，最基本的要求就是不让“误报”有可乘之机，导致合法流量也很有可能被意外拦截。如果触发了误报警报的流量恰好是来自上级、合作伙伴和客户的重要信息，IPS 不仅实施了一次性错误阻断，而且会切断与他们的信息通道，其结果不言而喻。

IPS 技术需要面对很多挑战，其中主要有三点：一是单点故障，二是性能瓶颈，三是误报和漏报。设计要求 IPS 必须以嵌入模式工作在网络中，而这就可能造成瓶颈问题或单点故障。如果 IDS 出现故障，最坏的情况也就是造成某些攻击无法被检测到，而嵌入式的 IPS 设备出现问题，就会严重影响网络的正常运转。如果 IPS 出现故障而关闭，用户就会面对一个由 IPS 造成的拒绝服务问题，所有客户都将无法访问企业网络提供的应用。

即使IPS设备不出现故障，它仍然是一个潜在的网络瓶颈，不仅会增加滞后时间，而且会降低网络的效率，IPS必须与数千兆或者更大容量的网络流量保持同步，尤其是当加载了数量庞大的检测特征库时，设计不够完善的IPS嵌入设备无法支持这种响应速度。绝大多数高端IPS产品供应商都通过使用自定义硬件（FPGA、网络处理器和ASIC芯片）来提高IPS的运行效率。

误报率和漏报率也需要IPS认真面对。在繁忙的网络当中，如果以每秒需要处理十条警报信息来计算，IPS每小时至少需要处理36000条警报，一天就是864000条。一旦生成了警报，最基本的要求就是IPS能够对警报进行有效处理。如果入侵特征编写得不是十分完善，那么“误报”就有了可乘之机，导致合法流量也有可能被意外拦截。对于实时在线的IPS来说，一旦拦截了“攻击性”数据包，就会对来自可疑攻击者的所有数据流进行拦截。如果触发了误报警报的流量恰好是某个客户订单的一部分，其结果可想而知，这个客户整个会话就会被关闭，而且此后该客户所有重新连接到企业网络的合法访问都会被“尽职尽责”的IPS拦截。

IPS厂商采用各种方式加以解决。一是综合采用多种检测技术，二是采用专用硬件加速系统来提高IPS的运行效率。尽管如此，为了避免IPS重蹈IDS覆辙，厂商对IPS的态度还是十分谨慎的。例如，NAI提供的基于网络的入侵防护设备提供多种接入模式，其中包括旁路接入方式，在这种模式下运行的IPS实际上就是一台纯粹的IDS设备，NAI希望提供可选择的接入方式来帮助用户实现从旁路监听向实时阻止攻击的自然过渡。

IPS的不足并不会成为阻止人们使用IPS的理由，因为安全功能的融合是大势所趋，入侵防护顺应了这一潮流。对于用户而言，在厂商提供技术支持的条件下，有选择地采用IPS，仍不失为一种应对攻击的理想选择。

### 2.4.9 IPS的产品种类

#### 1. 基于主机的入侵防护(HIPS)

HIPS（Host IPS）通过在主机/服务器上安装软件代理程序，防止网络攻击入侵操作系统以及应用程序。基于主机的入侵防护能够保护服务器的安全弱点不被不法分子所利用。Cisco公司的Okena、NAI公司的McAfee Entercept、冠群金辰的龙渊服务器核心防护都属于这类产品，因此它们在防范红色代码和Nimda的攻击中，起到了很好的防护作用。基于主机的入侵防护技术可以根据自定义的安全策略以及分析学习机制来阻断对服务器、主机发起的恶意入侵。HIPS可以阻断缓冲区溢出、改变登录口令、改写动态链接库以及其他试图从操作系统夺取控制权的入侵行为，整体提升主机的安全水平。

在技术上，HIPS采用独特的服务器保护途径，利用包过滤、状态包检测和实时入侵检测组成分层防护体系。这种体系能够在提供合理吞吐率的前提下，最大限度地保护服务器的敏感内容，既可以软件形式嵌入到应用程序对操作系统的调用当中，通过拦截针对操作系统的可疑调用，提供对主机的安全防护；也可以更改操作系统内核程序的方式，提供比操作系统更加严谨的安全控制机制。

由于HIPS工作在受保护的主机/服务器上，它不但能够利用特征和行为规则检测，阻止诸如缓冲区溢出之类的已知攻击，还能够防范未知攻击，防止针对Web页面、应用和资源的未授权的任何非法访问。HIPS与具体的主机/服务器操作系统平台紧密相关，不同的平台需要不同的软件代理程序。

### 2．基于网络的入侵防护(NIPS)

NIPS（Network IPS）通过检测流经的网络流量，提供对网络系统的安全保护。由于它采用在线连接方式，所以一旦辨识出入侵行为，NIPS 就可以去除整个网络会话，而不仅仅是复位会话。同样由于实时在线，NIPS 需要具备很高的性能，以免成为网络的瓶颈，因此 NIPS 通常被设计成类似于交换机的网络设备，提供线速吞吐速率以及多个网络端口。

NIPS 必须基于特定的硬件平台，才能实现千兆级网络流量的深度数据包检测和阻断功能。这种特定的硬件平台通常可以分为三类：一是网络处理器（网络芯片），二是专用的 FPGA 编程芯片，三是专用的 ASIC 芯片。

在技术上，NIPS 吸取了目前 NIDS 所有的成熟技术，包括特征匹配、协议分析和异常检测。特征匹配是最广泛应用的技术，具有准确率高、速度快的特点。基于状态的特征匹配不但检测攻击行为的特征，还要检查当前网络的会话状态，避免受到欺骗攻击。

协议分析是一种较新的入侵检测技术，它充分利用网络协议的高度有序性，并结合高速数据包捕捉和协议分析，来快速检测某种攻击特征。协议分析正在逐渐进入成熟应用阶段。协议分析能够理解不同协议的工作原理，以此分析这些协议的数据包，来寻找可疑或不正常的访问行为。协议分析不仅仅基于协议标准（如 RFC），还基于协议的具体实现，这是因为很多协议的实现偏离了协议标准。通过协议分析，IPS 能够针对插入（Insertion）与规避（Evasion）攻击进行检测。异常检测的误报率比较高，NIPS 不将其作为主要技术。

### 3．应用入侵防护(AIP)

NIPS 产品有一个特例，即应用入侵防护（Application Intrusion Prevention，AIP），它把基于主机的入侵防护扩展成位于应用服务器之前的网络设备。AIP 被设计成一种高性能的设备，配置在应用数据的网络链路上，以确保用户遵守设定好的安全策略，保护服务器的安全。NIPS 工作在网络上，直接对数据包进行检测和阻断，与具体的主机/服务器操作系统平台无关。

NIPS 的实时检测与阻断功能很有可能出现在未来的交换机上。随着处理器性能的提高，每一层次的交换机都有可能集成入侵防护功能。

## 2.5 网关级防护：UTM

随着技术的快速发展，安全方面出现了许多新的问题：①新攻击已逃避了传统的安全技术，防病毒系统只扫描有限数量的协议，例如邮件协议（SMTP、POP3、IMAP）、Web 协议（HTTP）、文件传输协议（FTP），新攻击采用防病毒系统未扫描的协议，例如 RPC、TFTP、SQL 等。②入侵防御系统（Intrusion Prevention Systems）需要手动方式更新，但新漏洞攻击传播很快，即称为“零日”（zero-day）或“零小时”（zero-hour）的攻击，手动方式更新 IPS 配置太慢。③混合攻击方式大量涌现，单一功能的防火墙远不能满足业务的需要。④攻击层面已经从传统的网络层数据攻击升级到多层次协议的复合型攻击。这些都极大增加了安全维护成本，用户的安全需求在从安全产品向整体解决方案转移，在边界阻挡威胁十分关键，因此 UTM 应运而生。

UTM 是英文 Unified Threat Management 的缩写，即统一威胁管理。它最早由 Fortinet（飞塔）公司在 2002 年提出的。2004 年 9 月，IDC（国际数据公司）在 31840 号报告中提出这个信息安

全概念。UTM 被列为 2005 年的十大热点技术之一。

UTM 是集网络防火墙、IDS/IPS（入侵检测/防御/阻断）、网关防病毒、VPN、Web/E-mail 内容过滤、防垃圾邮件、防内外网拒绝服务攻击（DoS）、访问控制、访问跟踪、NAT、内网监控（网址过滤）、防间谍软件功能、安全审计功能、数据中心功能、AAA 认证、电子证书及密钥管理等为一体的 All-In-One 安全网关，如图 2-10 所示。这样不需部署多种安全设备，只需一台设备全面保护网络安全。

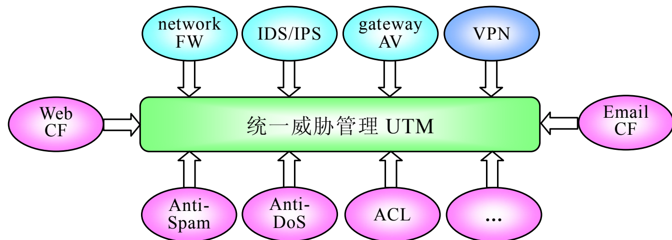



图 2-10　UTM 平台的综合功能

通过 UTM 平台综合功能图示我们不难看出，IDS/IPS 功能引擎是其中的核心功能部件，而网络异常检测技术则是该部件的核心技术。从技术的角度上讲，IDS/IPS 功能引擎中应注重解决下面三个问题。

（1）尽可能地减少对网络性能的影响。IDS/IPS 功能引擎入侵防御系统在计算量上被认为达到普通包过滤防火墙的 8～10 倍，如何在大流量和大计算量下保证吞吐量达到线速，延迟时间在可接受范围，对硬件设计和软件算法是极大挑战。

（2）对网络资源进行有针对性的防护，减少误报率，降低误报对正常网络应用的影响。入侵检测解决方案应该提供对攻击的实时预防和分析，在不同的应用环境下，能够有效进行针对当前网络资源的特定防护，并采用创新性的方法，降低当前的误报率。

（3）对新型攻击的自动免疫力。随着“网络钓鱼”、“僵尸网络”、蠕虫、病毒等网络威胁及其种类繁多的变种的出现，光靠不断扩充、完善漏洞库和特征库的方式来进行检测和防御很难跟上黑客的攻击步伐，目前市场上迫切需要基于行为检测技术、对未知攻击具有较强检测性能和免疫力的产品，那么以网络异常检测技术为核心的产品自然就成为其首选。

## 2.6 Web 应用综合防护：WAF

WAF（Web Application Firewall）即 Web 防火墙（见图 2-11），主要是针对 Web 特有入侵方式的加强防护，如 DDoS 防护、SQL 注入、XML 注入、XSS 等。由于是应用层而非网络层的入侵，从技术角度应该称为 Web IPS，而不是 Web 防火墙。这里之所以叫做 Web 防火墙，是因为大家比较好理解业界流行的称呼而已。由于重点是防 SQL 注入，也有人称为 SQL 防火墙。

Web 防火墙产品部署在 Web 服务器的前面，串行接入，不仅在硬件性能上要求高，而且不能影响 Web 服务，所以 HA 功能、Bypass 功能都是必须的，而且还要与负载均衡、Web Cache 等 Web 服务器前的常见的产品协调部署。

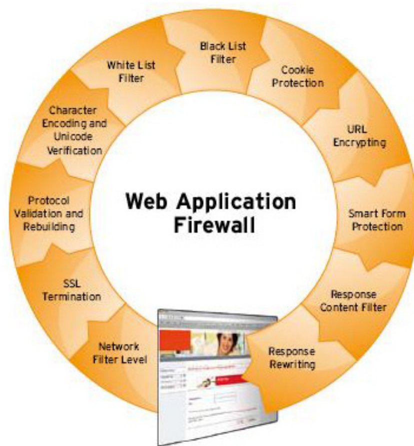



图 2-11　WAF

Web 防火墙的主要技术对入侵的检测能力，尤其是对 Web 服务入侵的检测，不同的厂家技术差别很大，不能以厂家特征库大小来衡量，主要是看测试效果，从厂家技术特点来说，有下面几种方式。

- 代理服务。代理方式本身就是一种安全网关，基于会话的双向代理，中断了用户与服务器的直接连接，适用于各种加密协议，这也是 Web 的 Cache 应用中最常用的技术。代理方式防止了入侵者的直接进入，对 DDoS 攻击可以抑制，对非预料的“特别”行为也有所抑制。Netcontinuum（梭子鱼）公司的 WAF 就是这种技术的代表。
- 特征识别。识别出入侵者是防护的前提。特征就是攻击者的“指纹”，如缓冲区溢出时的 Shellcode，SQL 注入中常见的“真表达（1=1）”……应用信息没有“标准”，但每个软件、行为都有自己的特有属性，病毒与蠕虫的识别就采用此方式，麻烦的就是每种攻击都有自己的特征，数量比较庞大，也极相似，误报的可能性也大。虽然目前恶意代码的特征以指数型增长，安全界声言要淘汰此项技术，但目前应用层的识别还没有特别好的方式。
- 算法识别。特征识别有缺点，人们在寻求新的方式。对攻击类型进行归类，相同类的特征进行模式化，不再是单个特征的比较，算法识别有些类似模式识别，但对攻击方式依赖性很强，如 SQL 注入、DDoS、XSS 等都开发了相应的识别算法。算法识别是进行语义理解，而不是靠“长相”识别。
- 模式匹配。它是 IDS 中“古老”的技术，把攻击行为归纳成一定模式，匹配后能确定是入侵行为，当然模式的定义有很深的学问，各厂家都隐秘为“专利”。协议模式是其中简单的，是按标准协议的规程来定义模式；行为模式就复杂一些。

Web 防火墙最大的挑战是识别率，这并不是一个容易测量的指标，因为漏网进去的入侵者，并非都大肆张扬，比如给网页挂马，你很难察觉进来的是哪一个，不知道当然也无法统计。对于已知的攻击方式，可以谈识别率；对未知的攻击方式，你也只好等它自己“跳”出来才知道。

Imperva 公司的 WAF 产品在提供入侵防护的同时，还提供了另外一个安全防护技术，就是对 Web 应用网页的自动学习功能，由于不同的网站不可能一样，所以网站自身页面的特性没有办法提前定义，所以 Imperva 采用设备自动预学习方式，从而总结出本网站的页面的特点。具体的做法如下。

通过一段时间的用户访问，WAF 记录了常用网页的访问模式，如一个网页中有几个输入点，输入的是什么类型的内容，通常情况的长度是多少……学习完毕后，定义出一个网页的正常使用模式，当今后有用户突破了这个模式，如一般的账号输入不应该有特殊字符，而 XML 注入时需要有“<”之类的语言标记，WAF 就会根据你预先定义的方式预警或阻断；再如密码长度一般不超过 20 位，在 SQL 注入时加入代码会很长，同样突破了网页访问的模式。

网页自学习技术，是从 Web 服务自身的业务特定角度入手，不符合常规就是异常的，也是入侵检测技术的一种，比起单纯的 Web 防火墙来，不仅给入侵者“下通缉令”，而且建立进入自家的内部“规矩”，这种双向的控制，显然比单向的要好。

Citrix 公司收购了 Teros 后，推出的应用防火墙通过分析双向流量来学习 Web 服务的用户行为模式，建立了若干用户行为模型，一旦匹配上你是某个行为，就按该模式行为去衡量你的行为做法，有“越轨”企图立即给予阻断。这个自适应学习引擎与 Imperva 公司的网页自学习有些类似，不过一个是重点学习网页特点，另一个是学习用户访问的规律。

# 2.7 数据防护：数据加密及备份

## 2.7.1 加密技术的基本概念

数据加密（encryption）是各类信息安全技术的基础。数据加密过程可以简单概括成：把数据 A 表达成 B；而反加密过程（破译或解密，decryption）是：把数据 B 恢复成 A，如图 2-12 所示。

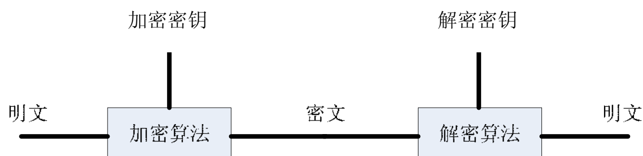



图 2-12 加解密技术原理示意图

数据加密的术语有：

（1）明文（plain text/clear text），指原始的或未加密的数据；

（2）密文（cipher text），指明文加密后的格式，是加密算法的输出信息，无密钥的用户无法理解，用于数据的存储以及传输；

（3）密钥（key）是一组编码信息，参与密码的运算，对密码运算起到特定的控制作用，密钥又可以进一步分为加密密钥（encryption key）和解密密钥（decryption key）。

## 2.7.2 加密系统的分类

### 1. 对称加密系统

对称加密是指：加密和解密时使用相同的密钥。如图 2-13，在对称密钥密码体制中，用于加密的密钥与用于解密的密钥完全相同。在对称密钥体制中，通常使用的加密算法比较简便、高效，密钥简短，破译也比较困难。但是传送和保管密钥是一个十分严峻的问题。

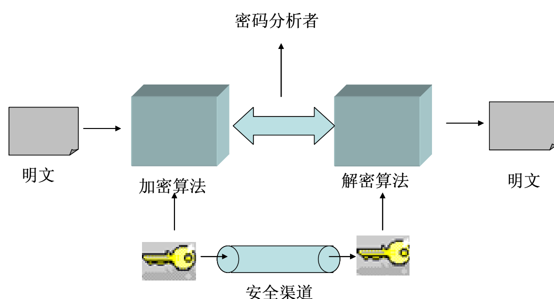



图 2-13 对称加密算法模型

对称加密体制存在着以下两个主要问题。

- 对称加密方法的主要问题是密钥的生成、注入、存储、管理、分发等很复杂，特别是随着用户的增加，密钥的需求量成倍增加。在网络通信中，大量密钥的分配是一个难以解决的问题。这种对称密钥体制中，加密与解密使用的是相同的密钥，收发双方都必须知道该密钥。因此，他们必须事先通过某种秘密途径商定密钥，如果没有事先约定密钥，则不可能与他人通信。另外，对于一个完备的通信网，需要的密码数量非常大，网上若有 n 个用户，则需要的密钥数为 C (n, 2) =n(n-1)/2 个，如 n=1000 时，所需的密钥数 C (1000, 2)约 499 500 个。这么多密钥的管理和必需的更换都将是十分繁重的工程。更有甚者，每个用户必须记下与其他 n-1 个用户通信所用的密钥。虽然以上问题都可以通过设置一个可信的集中的 KDC（Key Distribution Center，密钥分配中心）来

解决，但 KDC 所需要的通信量很大，而且需要在用户和 KDC 之间相互鉴别。因此，有关密钥的分配、安全传送、保密等管理是一件相当困难的事情。

- 数据的完整性保护方面的困难，不具备签名功能。在现实生活中，合同和协议的真实性是由书面签字来保证的，签字可以作为法律上的证据。但是在电子通信中要用技术手段来解决证实问题。一方面是指对接收方的证实，另一方面是指对发送方的证实，也就是能够确认接收方所收到和保存的信息确实是由发送方发出的，不是伪造的，也没有经过任何人（包括接收方在内）的篡改。在对称密码体制中，接收方利用保密密钥对密文信息进行解密变换，能对发送方进行证实，其他人不能够伪造发送方的信息。但是对称密码却无法解决对接收方的证实问题。因为接收方所掌握的解密密钥和发送方的加密密钥相同，完全有能力篡改他接受到的文件或伪造文件，也就是说，接收方可以伪造一份对他有利的明文，用同样的密钥加密，然后诬陷说这份伪造的明文来自发送方。基于同样的理由发送方完全有借口否定他曾发出的明文，因为接收方有可能篡改明文。因此，对称加密体制在数字签名和身份验证方面的应用显然是比较困难的。

### 2. 非对称加密系统

非对称加密是指：加密和解密使用不同的密钥。与对称加密算法不同，非对称加密算法需要两个密钥：公开密钥（public key）和私有密钥（private key）。公开密钥与私有密钥是一对，如果用公开密钥对数据进行加密，只有用对应的私有密钥才能解密；如果用私有密钥对数据进行加密，那么只有用对应的公开密钥才能解密。因为加密和解密使用的是两个不同的密钥，所以这种算法叫做非对称加密算法，如图 2-14 所示。

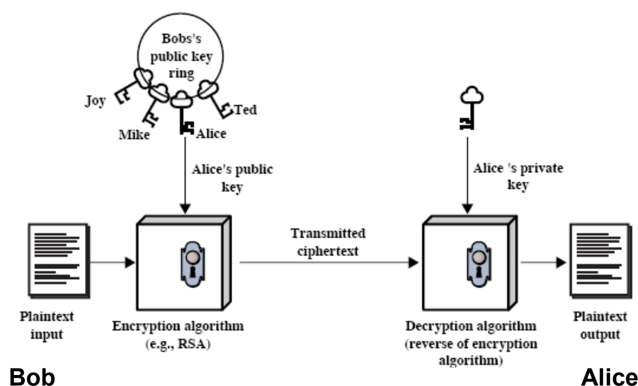


图 2-14　非对称加密过程示例

非对称加密算法实现机密信息交换的基本过程是：甲方生成一对密钥并将其中的一把作为公用密钥向其他方公开；得到该公用密钥的乙方使用该密钥对机密信息进行加密后再发送给甲方；甲方再用自己保存的另一把私有密钥对加密后的信息进行解密。反之亦然。非对称加密算法的保密性比较好，它消除了最终用户交换密钥的需要，但加密和解密花费时间长、速度慢，其不适合于对文件加密而只适用于对少量数据进行加密。

非对称加密算法中的加密密钥和解密密钥是不一样的。要找到一种非对称加密算法是一件很困难的事，Ron Rivest、Adi Shamir 和 Leonard Adleman 终于在 1978 年提出了 RSA 公开密钥算法，是现在应用最广泛的一种非对称加密算法，这种算法的运算非常复杂，速度也很慢。

假如你的朋友需要发送一些非常重要、非常机密的信息给你，而你跟外界的每一条通路都被监听了。有人认为，你的朋友可以用 DES 对称加密算法对信息加密后传送给你，可是密钥也是需要传送的，一旦密钥被截获，后果可想而知。如果说经常跟你通信的朋友还可以事先跟你约定好密钥，那么在浩瀚的 Internet 上那么多人和机构是没有办法跟你事先就约定好的。

非对称加密系统（也就是公开密钥系统）的作用就在于，此时，你可以先将公钥（加密密钥）发布在一个可靠的第三方网站上，让你的朋友获得这个公钥，然后用此公钥对信息加密后传送给你，你再用解密密钥（私钥）恢复信息的明文进行阅读，在这个过程中私钥（解密密钥）不会以任何形式传送，只掌握在你的手中，也就是说你的朋友对信息加密后，他自己也没办法再解开进行验证。监听者得到了加密密钥，却无法得出解密密钥，也就无法查看信息的明文。

3．哈希 Hash 算法

Hash 算法特别的地方在于它是一种单向算法，用户可以通过 Hash 算法对目标信息生成一段特定长度的唯一的 Hash 值，却不能通过这个 Hash 值重新获得目标信息。因此 Hash 算法常用在不可还原的密码存储、信息完整性校验等。常见的 Hash 算法有 MD2、MD4、MD5、HAVAL、SHA。

### 2.7.3 常用的加密算法

1．对称加密算法：DES 算法

DES 是 Data Encryption Standard（数据加密标准）的缩写，是由 IBM 公司研制的一种加密算法，属于秘密密钥算法，同时也是一种对称密钥算法，即加密和解密采用同种算法，密钥是同一把，接收方和发送方都执有相同的密钥。DES 算法采用专门的变换函数来加密明文。方法是先把明文按每组长 64 位分成若干组，然后用变换函数依次加密，每次输出 64 位的密文，最后将所有密文串接即得整个密文。密钥长度为 56 位（每个第 8 位都用作奇偶校验），可以是任意 56 位的数，且可以在任意时候改变。DES 的安全性完全依赖于对密钥的保护（故称秘密密钥算法），泄漏密钥意味着任何人都能对加密信息进行解密，因此，在公开的计算机网络上安全传送和保管密钥是一个严峻的问题。但 DES 算法运算速度快、稳定，适合对大量数据的加密。

2．对称加密算法：3DES 算法

3DES（即 Triple DES）是 DES 向 AES 过渡的加密算法，它使用 3 条不同的 64 位的密钥对数据进行三次加密。它是 DES 的一个更安全的变形。它以 DES 为基本模块，通过组合分组方法设计出分组加密算法，比起最初的 DES、3DES 更为安全。

3．非对称加密算法：RSA 算法

RSA 算法于 1977 年由美国麻省理工学院（MIT）的 Ronal Rivest、Adi Shamir 和 Len Adleman 3 位年轻教授提出，并以 3 人的姓氏命名为 RSA。该算法利用数论领域的一个事实，即把两个大质数相乘生成 1 个合数是件十分容易的事情，但要把 1 个合数分解为两个质数却十分困难。目前合数分解问题仍然是数学领域尚未解决的一大难题，至今没有任何高效的分解方法。与 DES 算法相比，RSA 算法具有明显的优越性，因为它无须收发双方同时参与加密过程，密钥以成对方式出现，一把保密而另一把公开（故称为公开密钥算法）。作为一种非对称密钥算法，RSA 算法的加密数据的密钥和解密数据的密钥不同，以易于密钥的安全分发，适合于在电子函件系统、数字签名等的加密。RSA 算法的缺点是运算速度太慢，比 DES 算法慢几个数量级，因此只适合对少量关键数据的加密。

4．非对称加密算法：DSA 算法

Digital Signature Algorithm（DSA）是 Schnorr 和 ElGamal 签名算法的变种，被美国 NIST 作为 DSS（Digital Signature Standard）。DSA 是基于整数有限域离散对数难题的，其安全性与 RSA 相比差不多。DSA 的一个重要特点是两个素数公开，这样，当使用别人的 p 和 q 时，即使不知道私钥，你也能确认它们是否是随机产生的，还是作了手脚。RSA 算法却作不到。

5．单向散列算法：MD5 算法

MD5 的全称是 Message-Digest Algrithm5（第 5 类报文摘要），于 20 世纪 90 年代初由 MIT 的计算机科学实验室和 RSADataSecurityInc 发明，经 MD2、MD3 和 MD4 发展而来。MD5 用 Hash（哈希）函数来加密明文，但产生的密文无法恢复成明文，是一种单向的加密算法，因此不能用于数据的保密。其加密方法是先把明文 m 进行 Hash 运算后产生一小段固定长度的报文摘要 H（m），并将其附在明文后发送，接收方收到后也用 MD5 算法对明文 m 进行 Hash 运算，得到 H′（m），如果 H′（m）=H（m），即认为报文是对方发送的，而且在传输过程中没有被篡改过。因此，使用 MD5 时，明文是不加密的，也没有加密的必要，因为明文内容一般是公告信息，使用 MD5 的目的仅是保证公告的权威性，类似于现实生活中在公告文件上加盖公章。基于上述特点，MD5 算法主要用于防止篡改或伪造报文。

6．单向散列算法：SHA1 算法

SHA（Secure Hash Algorithm）：可以对任意长度的数据运算生成一个 160 位的数值；在 1993 年，安全散列算法（SHA）由美国国家标准和技术协会（NIST）提出，并作为联邦信息处理标准（FIPS PUB 180）公布；1995 年又发布了一个修订版 FIPS PUB 180-1，通常称之为 SHA-1。SHA-1 是基于 MD4 算法的，并且它的设计在很大程度上是模仿 MD4 的。现在已成为公认的最安全的散列算法之一，并被广泛使用。

### 2.7.4 加密算法的主要应用场景

随着密码学商业应用的普及，公钥密码学受到前所未有的重视。除传统的密码应用系统外，PKI（Public Key Infrastructure）系统以非对称公钥密码技术为主，提供加密、签名、认证、密钥管理、分配等功能。

- 保密通信。保密通信是密码学产生的动因。使用公、私钥密码体制进行保密通信时，信息接收者只有知道对应的密钥才可以解密该信息。
- 数字签名。数字签名技术可以代替传统的手写签名，而且从安全的角度考虑，数字签名具有很好的防伪造功能。在政府机关、军事领域、商业领域有广泛的应用环境。
- 秘密共享。秘密共享技术是指将一个秘密信息利用密码技术分拆成 n 个称为共享因子的信息，分发给 n 个成员，只有 k（k≤n）个合法成员的共享因子才可以恢复该秘密信息，其中任何一个或 m（m≤k）个成员合作都不知道该秘密信息。利用秘密共享技术可以控制任何需要多个人共同控制的秘密信息、命令等。
- 认证功能。在公开的信道上进行敏感信息的传输，采用签名技术实现对消息的真实性、完整性进行验证，通过验证公钥证书实现对通信主体的身份验证。
- 密钥管理。密钥是保密系统中更为脆弱而重要的环节，公钥密码体制是解决密钥管理工作的有力工具；利用公钥密码体制进行密钥协商和产生，保密通信双方不需要事先

共享秘密信息；利用公钥密码体制进行密钥分发、保护、密钥托管、密钥恢复等。

基于公钥密码体制可以实现以上通用功能以外，还可以设计实现以下的系统：安全电子商务系统、电子现金系统、电子选举系统、电子招投标系统、电子彩票系统等。

### 2.7.5 数据备份及恢复技术

备份不仅是数据的保护，其最终目的是为了在系统遇到人为或自然灾难时，能够通过备份内容对系统进行有效的灾难恢复。备份不仅是单纯的拷贝，管理也是备份重要的组成部分。管理包括备份的可计划性、磁带机的自动化操作、历史记录的保存以及日志记录等。对于现代企业来说，在正常运作的过程中难免出现如下几种情况的灾难，需要针对它们进行数据的备份和恢复工作。

- 计算机软硬件故障：对于某一企业，发生的可能性最大，也最频繁，是经常发生的一类故障。
- 人为操作故障：对管理较严、人员素质较高的企业，偶尔发生；对管理松散、人员培训不足的企业，会经常发生。
- 资源不足引起的计划性停机：对于某一企业，随着业务的快速增长，平均每年均会发生如软、硬件升级，系统资源扩充等事件，业务增长越快的企业，发生亦越频繁。
- 自然灾害：由于火灾、水灾、地震、雷击等自然灾害引起的企业计算资源和数据的毁灭或者丢失。这种情况对于任何企业来说都是不可避免和抗拒的，当然发生的概率相对上述几种灾害来说通常比较小。

根据目前的应用情况来看，如下一些分类的备份方法可以提供给企业根据不同的应用情况进行选用。

1）备份模式

- 逻辑备份：每个文件都是由不同的逻辑块组成。每一个逻辑的文件块存储在连续的物理磁盘块上，但组成一个文件的不同逻辑块极有可能存储在分散的磁盘块上。备份软件通常既可以进行文件操作，又可以对磁盘块进行操作。基于文件的备份系统能够识别文件结构，并拷贝所有的文件和目录到备份资源上。这样的系统跨越了存储在每个inode上的指针，可顺序地读取每个文件的物理块，然后备份软件连续地将文件写入到备份媒介上。这样的备份使得每个单独文件的恢复变得很快，但连续的存储文件会使得备份速度减慢，因为在对非连续存储磁盘上的文件进行备份时需要额外的查找操作。这些额外的操作增加了磁盘的开销，降低了磁盘的吞吐率。另外，对于文件，即使一个很小的改变，基于文件的逻辑备份也需将整个文件备份。
- 物理备份：系统在拷贝磁盘块到备份媒介上时忽略文件结构，这会提高备份的性能，因为备份软件在执行过程中，花费在搜索操作上的开销很少。但这种方法使得文件的恢复变得复杂且缓慢，因为文件并不是连续地存储在备份媒介上。为了允许文件恢复，基于设备的备份必须要收集文件和目录是如何在磁盘上组织信息的，才能使备份媒介上的物理块与特定的文件相关联。因而，基于设备的备份适合于指定一个特定的文件系统来实现，并且不易移植。而基于文件的方案则更易移植，因为备份文件包含的是连续文件。另外，基于设备的备份方案可能会导致数据的不一致。

2）备份策略

- 全备份：这种备份方式很直观，容易被人理解。当发生数据丢失的灾难时，只要用一

盘磁带（即灾难发生前一天的备份磁带），就可以恢复丢失的数据。但也存在不足之处：首先，每天都对系统进行完全备份，在备份数据中有大量内容是重复的，例如操作系统与应用程序，这些重复的数据占用了大量的磁带空间，意味着增加成本；另外，由于需要备份的数据量相当大，备份所需的时间也就较长。

- 增量备份：该备份的优点是没有重复的备份数据，节省磁带空间，缩短备份时间。缺点在于当发生灾难时，恢复数据比较麻烦。例如，若系统在周四早晨发生故障，那么就需要将系统恢复到周三晚上的状态。管理员需要找出周一的完全备份磁带进行系统恢复，再找出周二的磁带来恢复星期二的数据，最后再找出周三的磁带来恢复星期三的数据。在这种备份下，各磁带间的关系就像链子一样，其中任何一盘磁带出了问题，都会导致整条链子脱节。
- 差分备份：管理员先在周一进行一次系统完全备份，然后在接下来的几天里，再将当天所有与星期一不同的数据备份到磁带上。差分备份无须每天都做系统完全备份，备份所需时间短，节省磁带空间，灾难恢复也很方便。系统管理员只需两盘磁带，即系统全备份的磁带与发生灾难前一天的备份磁带，就可以将系统完全恢复。

3）备份服务器在备份过程中是否可接收用户响应和数据更新角度

- 冷备份：冷备份很好地解决了在备份选择进行时并发更新带来的数据不一致性问题，缺点是用户需要等待很长的时间，服务器将不能及时响应用户的需求。目前的新技术有 LAN-Free、Server-Free 等，这种方式的恢复时间比较长，但投资较少。
- 热备份：由于是同步备份，热备份资源占用比较多，投资较大，但是它的恢复时间非常短。在热备份中有一个很大的问题就是数据的有效性和完整性，如果备份过程中产生了数据不一致性的问题，会导致数据的不可用。解决此问题的方法是对于一些总是处于打开状态的重要数据文件，备份系统可以采取文件的单独写/修改特权，保证在该文件备份期间其他应用不能对它进行更新。

4）备份地点

- 远程备份：在本地将关键数据备份，然后送到异地保存，以防止企业本地备份的数据在灾难中也遭到破坏。灾难发生后，按预定数据恢复程序恢复系统和数据。这种方案可以采用磁带机、磁盘阵列等存储设备进行本地备份，同样还可以选择磁带库、光盘库等存储设备。
- 本地备份：在本地将关键数据进行备份，然后进行保存，主要包括本地双机备份、本地局域网备份等。

5）备份介质

- 磁带备份：磁带一开始就是储存恢复备份的主要介质，随着容量和速度的不断提升，磁带可以让用户以较低的成本存储多重备份或版本。由于它是一种可移动式的介质，所以也可以作为远程灾难恢复备份用途。磁带备份最大的挑战是，其备份质量并不稳定。在操作过程中，有可能数据已经全部备份成功，但是，却很难验证磁带内所有数据是否皆可恢复。劣质的磁带会令恢复操作失败，而这种错误用户通常很难察觉，直到执行恢复操作时才会发现，但那时已经来不及了。当备份数据量非常大时，价格优势十分明显。
- 磁盘备份：首先，磁盘阵列不需要稳定的数据流，即使采用只存储少量数据的增量备

份，也没有“摩擦”效应。其次，磁盘阵列允许管理人员进行较不常做的全部数据备份工作（Full Backup），而不用忍受执行速度变慢的后果或增加恢复资料时损坏的风险，可以简化及加快整体备份的速度。除了缩短备份时间外，对于时常进行的完整备份，也可以减少恢复时所需的磁带数量，简化恢复过程。以磁带备份来看，恢复时所需的磁带数量会依增量备份的次数而增加，而磁盘阵列允许管理人员简化这样的备份过程。在制作远程恢复备份时，磁盘也比较容易且更有效。利用“磁盘至磁带”复制数据时，不需要同时由一个磁带内处理多位使用者的恢复备份，它可以针对每位使用者直接复制恢复备份，因此加快了恢复的过程。另外，“磁盘至磁带”也比“磁带至磁带”复制方式更有弹性。当正在进行“磁带至磁带”复制时，主要来源是磁带和要备份的磁带，两种都没有办法再做其他的备份或恢复操作。相反的，当正在复制数据到磁带时，磁盘允许使用者同时存取数据，接受备份及恢复操作。

灾难恢复措施在整个备份制度中占有相当重要的地位，因为它关系到系统在经历灾难后能否迅速恢复。灾难恢复操作通常可以分为两类：第一类是全盘恢复；第二类是个别文件恢复。还有一种值得一提的是重定向恢复。

- 全盘恢复：全盘恢复一般应用在服务器发生意外灾难导致数据全部丢失、系统崩溃或是有计划的系统升级、系统重组等，也称为系统恢复。
- 个别文件恢复：由于操作人员的水平不高，个别文件恢复可能要比全盘恢复常见得多，利用网络备份系统的恢复功能，我们很容易恢复受损的个别文件。只需浏览备份数据库或目录，找到该文件，触动恢复功能，软件将自动驱动存储设备，加载相应的存储媒体，然后恢复指定文件。
- 重定向恢复：重定向恢复是将备份的文件恢复到另一个不同的位置或系统上去，而不是进行备份操作时它们当时所在的位置。重定向恢复可以是整个系统恢复，也可以是个别文件恢复。重定向恢复时需要慎重考虑，要确保系统或文件恢复后的可用性。为了防备数据丢失，我们需要做好详细的灾难恢复计划，同时还要定期进行灾难演练。每过一段时间，应进行一次灾难演习。可以利用淘汰的机器或多余的硬盘进行灾难模拟，以熟练灾难恢复的操作过程，并检验所生成的灾难恢复软盘和灾难恢复备份是否可靠。

## 2.8 远程访问安全保障：VPN

### 2.8.1 VPN 简介

VPN 是英文 Virtual Private Network 的缩写，中文译为虚拟专用网，如图 2-15 所示。VPN 是利用公共网络基础设施，通过“隧道”技术等手段达到类似私有专网的数据安全传输。我们通常将 VPN 当作 WAN 解决方案。但它也可以简单地用于 LAN。VPN 类似于点到点直接拨号连接或租用线路连接，尽管它是以交换和路由的方式工作。可以说，VPN 是 Intranet（内部网）在公众网络上的延伸，它可以提供与专用网一样的安全性、可管理性和传输性能，而建设、运转和维护网络的工作也从企业内部的 IT 部门剥离出来，交由运营商来负责。VPN 是建立在实际网络（或物理网络）基础上的一种功能性网络，它利用公共网络作为企业骨干网的低成本优势，同时克服

公共网络缺乏保密性的弱点。在 VPN 网络中，位于公共网络两端的网络在公共网络上传输信息时，其信息都是经过安全处理的，可以保证数据的完整性、真实性和私有性。VPN 是指在共用网络上建立专用网络的技术，之所以称为虚拟网主要是因为整个 VPN 网络的任意两个结点之间的连接并没有传统专网建设所需的点到点的物理链路，而是架构在公用网络服务商 ISP 所提供的网络平台之上的逻辑网络。用户的数据是通过 ISP 在公共网络（Internet）中建立的逻辑隧道（Tunnel），即点到点的虚拟专线进行传输的。通过相应的加密和认证技术来保证用户内部网络数据在公网上安全传输，从而真正实现网络数据的专有性。VPN 是企业网在因特网等公共网络上的延伸，它能在公共网络上创建一个安全的私有连接，因此使公司的远程用户、分支机构、业务伙伴等与公司的企业网连接起来，构成一个扩展的企业网。

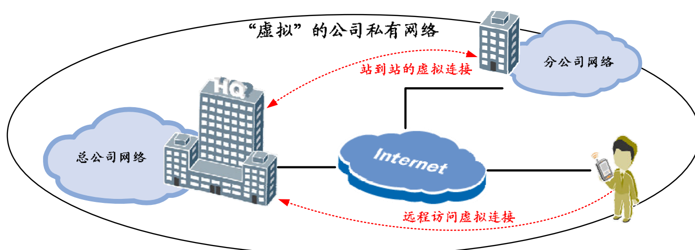


图 2-15　VPN 示意图

虚拟专用网可以帮助远程用户、公司分支机构、商业伙伴及供应商同公司的内部网建立可信的安全连接，并保证数据的安全传输。一个企业的虚拟专用网解决方案将大幅度地减少用户花费在城域网和远程网络连接上的费用。同时，这将简化网络的设计和管理，加速连接新的用户和网站。另外，虚拟专用网还可以保护现有的网络投资。随着用户的商业服务不断发展，企业的虚拟专用网解决方案可以使用户将精力集中到自己的生意上，而不是网络上。虚拟专用网可用于不断增长的移动用户的全球因特网接入，以实现安全连接；可用于实现企业网站之间安全通信的虚拟专用线路，用于经济有效地连接到商业伙伴和用户的安全外联网虚拟专用网。

虚拟专用网至少应能提供以下功能。

- 加密数据：以保证通过公网传输的信息即使被他人截获也不会泄露。
- 信息认证和身份认证：保证信息的完整性、合法性，并能鉴别用户的身份。
- 提供访问控制：不同的用户有不同的访问权限。

VPN 具有以下优点。

（1）降低成本：企业不必租用长途专线建设专网，无须大量的网络维护人员和设备投资。利用现有的公用网组建的 Intranet，要比租用专线或铺设专线节省开支，而且当距离越远时节省得越多。例如，企业的北京与纽约分部之间的连接，不太可能自铺专线；当一个远程用户在纽约想要连接到北京的 Intranet，用拨号访问时，花的是国际长途话费；而使用 VPN 技术时，只需在纽约和北京分别连接到当地的 Internet 就实现了互联，双方花的都是市话费。

（2）容易扩展：网络路由设备配置简单，无须增加太多的设备，省时省钱。对于发展很快的企业来说，VPN 就更是不可不用了。如果企业组建自己的专用网，在扩展网络分支时，要考虑到网络的容量，架设新链路，增加互联设备，升级设备等；而实现了 VPN 就方便多了，只需连接到公用网上，对新加入的网络终端在逻辑上进行设置即可，无须考虑公用网的容量、设备问题等。

（3）完全控制主动权：VPN 上的设施和服务完全掌握在企业手中。例如，企业可以将拨号访问交给 NSP 去做，而由自己负责用户的查验、访问权、网络地址、安全性和网络变化管理等重要工作。

## 2.8.2 VPN的分类

VPN的分类方式比较混乱。不同的生产厂家在销售它们的VPN产品时使用了不同的分类方式，它们主要是从产品的角度来划分的。不同的ISP在开展VPN业务时也推出了不同的分类方式，他们主要是从业务开展的角度来划分的。而用户往往也有自己的划分方法，主要是根据自己的需求来进行的。下面简单介绍从不同角度对VPN的分类。

### 1．按接入方式划分

这是用户和运营商最关心的VPN划分方式。一般情况下，用户可能是专线上网的，也可能是拨号上网的，这要根据用户的具体情况而定。建立在IP网上的VPN也就对应的有两种接入方式：专线接入方式和拨号接入方式。

（1）专线VPN：它是为已经通过专线接入ISP边缘路由器的用户提供的VPN解决方案。这是一种“永远在线”的VPN，可以节省传统的长途专线费用。

（2）拨号VPN（又称VPDN）：它是向利用拨号PSTN或ISDN接入ISP的用户提供的VPN业务。这是一种“按需连接”的VPN，可以节省用户的长途电话费用。需要指出的是，因为用户一般是漫游用户，是按需连接的，因此VPDN通常需要做身份认证（比如利用CHAP和RADIUS）。

### 2．按协议实现类型划

这是VPN厂商和ISP最为关心的划分方式。根据分层模型，VPN可以在第二层建立，也可以在第三层建立（甚至有人把在更高层的一些安全协议也归入VPN协议）。

（1）第二层隧道协议：包括点到点隧道协议（PPTP）、第二层转发协议（L2F）、第二层隧道协议（L2TP）、多协议标记交换（MPLS）等。

（2）第三层隧道协议：包括通用路由封装协议（GRE）、IP安全（IPSec），这是目前最流行的两种三层协议。

第二层和第三层隧道协议的区别主要在于用户数据在网络协议栈的第几层被封装，其中GRE、IPSec和MPLS主要用于实现专线VPN业务，L2TP主要用于实现拨号VPN业务（但也可以用于实现专线VPN业务），当然这些协议之间本身不是冲突的，而是可以结合使用的。

### 3．按VPN的发起方式划分

这是客户和IPS最为关心的VPN分类。VPN业务可以是客户独立自主实现的，也可以是由ISP提供的。

（1）发起（也称基于客户的）：VPN服务提供的其始点和终止点是面向客户的，其内部技术构成、实施和管理对VPN客户可见，需要客户和隧道服务器（或网关）方安装隧道软件。客户方的软件发起隧道，在公司隧道服务器处终止隧道，此时ISP不需要做支持建立隧道的任何工作。经过对用户身份符（ID）和口令的验证，客户方和隧道服务器极易建立隧道。双方也可以用加密的方式通信。隧道一经建立，用户就会感觉到ISP不再参与通信。

（2）服务器发起（也称客户透明方式或基于网络的）：在公司中心部门或ISP处（POP，Point of Presence）安装VPN软件，客户无须安装任何特殊软件。主要为ISP提供全面管理的VPN服务，服务提供的起始点和终止点是ISP的POP，其内部构成、实施和管理对VPN客户完全透明。

在上面介绍的隧道协议中，目前 MPLS 只能用于服务器发起的 VPN 方式。

#### 4．按 VPN 的服务类型划分

根据服务类型，VPN 业务大致分为三类：接入 VPN（Access VPN）、内联网 VPN（Intranet VPN）和外联网 VPN（Extranet VPN）。通常情况下内联网 VPN 是专线 VPN。

（1）接入 VPN。这是企业员工或企业的小分支机构通过公网远程访问企业内部网络的 VPN 方式。远程用户一般是一台计算机，而不是网络，因此组成的 VPN 是一种主机到网络的拓扑模型。需要指出的是，接入 VPN 不同于前面的拨号 VPN，这是一个容易发生混淆的地方，因为远程接入可以是专线方式接入的，也可以是拨号方式接入的。

（2）内联网 VPN。这是企业的总部与分支机构之间通过公网构筑的虚拟网，这是一种网络到网络以对等的方式连接起来所组成的 VPN。

（3）外联网 VPN。这是企业在发生收购、兼并或企业间建立战略联盟后，使不同企业间通过公网来构筑的虚拟网。这是一种网络到网络以不对等的方式连接起来所组成的 VPN（主要在安全策略上有所不同）。

#### 5．按承载主体划分

营运 VPN 业务的企业既可以自行建设他们的 VPN 网络，也可以把此业务外包给 VPN 商。这是客户和 ISP 最关心的问题。

（1）自建 VPN。这是一种客户发起的 VPN。企业在驻地安装 VPN 的客户端软件，在企业网边缘安装 VPN 网关软件，完全独立于运营商建设自己的 VPN 网络，运营商不需要做任何对 VPN 的支持工作。企业自建 VPN 的好处是可以直接控制 VPN 网络，与运营商独立，并且 VPN 接入设备也是独立的。但缺点是 VPN 技术非常复杂，这样组建的 VPN 成本很高，QoS 也很难保证。

（2）外包 VPN。企业把 VPN 服务外包给运营商，运营商根据企业的要求规划、设计、实施和运维客户的 VPN 业务。企业可以因此降低组建和运维 VPN 的费用，而运营商也可以因此开拓新的 IP 业务增值服务市场，获得更高的收益，并提高客户的保持力和忠诚度。笔者将目前的外包 VPN 划分为两种：基于网络的 VPN 和基于 CE（用户边缘设备）的管理型 VPN（Managed VPN）。基于网络的 VPN 通常在运营商网络的呈现点（POP）安装电信级 VPN 交换设备。基于 CE 的管理型 VPN 业务是一种受信的第三方负责设计企业所希望的 VPN 解决方案，并代表企业进行管理，所使用的安全网关（防火墙、路由器等）位于用户一侧。

#### 6．按 VPN 业务层次模型划分

这是根据 ISP 向用户提供的 VPN 服务工作在第几层来划分的（注意不是根据隧道协议工作在哪一层划分的）。

（1）拨号 VPN 业务（VPDN）。这是第一种划分方式中的 VPDN（事实上是按接入方式划分的，因为很难明确 VPDN 究竟属于哪一层）。

（2）虚拟租用线（VLL）。这是对传统的租用线业务的仿真，用 IP 网络对租用线进行模拟，而从两端的用户看来，这样一条虚拟租用线等价于过去的租用线。

（3）虚拟专用路由网（VPRN）业务。这是对第三层 IP 路由网络的一种仿真。可以把 VPRN 理解成第三层 VPN 技术。

（4）虚拟专用局域网段（VPLS）。这是在 IP 广域网上仿真 LAN 的技术。可以把 VPLS 理

解成一种第二层 VPN 技术。

## 2.9 身份认证技术

身份认证技术是在计算机网络中确认操作者身份的过程而产生的解决方法。计算机网络世界中一切信息包括用户的身份信息都是用一组特定的数据来表示的，计算机只能识别用户的数字身份，所有对用户的授权也是针对用户数字身份的授权。如何保证以数字身份进行操作的操作者就是这个数字身份的合法拥有者，也就是说保证操作者的物理身份与数字身份相对应，身份认证技术就是为了解决这个问题，作为防护网络资产的第一道关口，身份认证有着举足轻重的作用。

在真实世界，对用户的身份认证基本方法可以分为以下三种。

（1）根据你所知道的信息来证明你的身份（what you know，你知道什么）。

（2）根据你所拥有的东西来证明你的身份（what you have，你有什么）。

（3）直接根据独一无二的身体特征来证明你的身份（who you are，你是谁），比如指纹、面貌等。

在网络世界，其中手段与真实世界中一致，为了达到更高的身份认证安全性，某些场景会将上面 3 种挑选 2 种混合使用，即所谓的双因素认证。

下面为大家介绍几种常见的认证形式。

### 2.9.1 静态密码

用户的密码是由用户自己设定的。在网络登录时输入正确的密码，计算机就认为操作者就是合法用户。实际上，由于许多用户为了防止忘记密码，经常采用诸如生日、电话号码等容易被猜测的字符串作为密码，或者把密码抄在纸上放在一个自认为安全的地方，这样很容易造成密码泄露。如果密码是静态的数据，在验证过程中，在计算机内存中和传输过程中可能会被木马程序或网络截获。因此，静态密码机制无论是使用还是部署都非常简单，但从安全性上来讲，用户名、密码方式是一种不安全的身份认证方式。它利用的是 what you know 方法。

### 2.9.2 智能卡（IC 卡）

智能卡是一种内置集成电路的芯片，芯片中存有与用户身份相关的数据，由专门的厂商通过专门的设备生产，是不可复制的硬件，如图 2-16 所示。智能卡由合法用户随身携带，登录时必须将智能卡插入专用的读卡器读取其中的信息，以验证用户的身份。

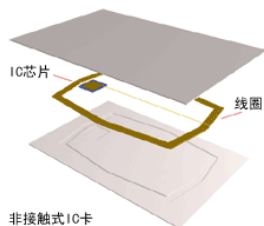


图 2-16 IC 卡

智能卡认证是通过智能卡硬件不可复制来保证用户身份不会被仿冒。然而由于每次从智能卡中读取的数据是静态的，通过内存扫描或网络监听等技术还是很容易截取到用户的身份验证信息，因此仍然存在安全隐患。它利用的是 what you have 方法。

### 2.9.3 短信密码

短信密码以手机短信形式请求包含 6 位随机数的动态密码，身份认证系统以短信形式发送随机的 6 位密码到客户的手机上。客户在登录或者交易认证时输入此动态密码，从而确保系统身份认证的安全性。它利用的是 what you have 方法。

短信密码具有以下优点。

（1）安全性

由于手机与客户绑定比较紧密，短信密码的生成与使用场景是物理隔绝的，因此密码在通路上被截取的几率降至最低。

（2）普及性

只要会接收短信即可使用，大大降低了短信密码技术的使用门槛，学习成本几乎为零，所以在市场接受度上面不会存在阻力。

（3）易收费

由于移动互联网用户天然养成了付费的习惯，这和 PC 时代互联网是截然不同的理念，而且收费通道非常发达，如果是网银、第三方支付、电子商务可将短信密码作为一项增值业务，每月通过 SP 收费不会有阻力，因此也可增加收益。

（4）易维护

由于短信网关技术非常成熟，大大降低了短信密码系统上马的复杂度和风险，短信密码业务后期客服成本低，稳定的系统在提升安全的同时也营造了良好的口碑效应，这也是目前银行大量采纳这项技术很重要的原因。

### 2.9.4 动态口令牌

目前最为安全的身份认证方式，也利用 what you have 方法，是一种动态密码。

动态口令牌（见图 2-17）是客户手持用来生成动态密码的终端，主流的是基于时间同步方式的，每 60 秒变换一次动态口令，口令一次有效，它产生 6 位动态数字进行一次一密的方式认证。




图 2-17 动态口令牌

由于它使用起来非常便捷，85%以上的世界 500 强企业运用它保护登录安全，广泛应用在 VPN、网上银行、电子政务、电子商务等领域。

### 2.9.5 USB Key

基于 USB Key 的身份认证方式是近几年发展起来的一种方便、安全的身份认证技术。它采用软硬件相结合、一次一密的强双因子认证模式，很好地解决了安全性与易用性之间的矛盾。USB Key（见图 2-18）是一种 USB 接口的硬件设备，它内置单片机或智能卡芯片，可以存储用户的密钥或数字证书，利用 USB Key 内置的密码算法实现对用户身份的认证。基于 USB Key 身份



图 2-18 USB Key

认证系统主要有两种应用模式：一是基于冲击/响应的认证模式，二是基于 PKI 体系的认证模式，目前运用在电子政务、网上银行等。

### 2.9.6 生物识别技术

生物识别是运用 who you are 方法，通过可测量的身体或行为等生物特征进行身份认证的一种技术。生物特征是指唯一的可以测量或可自动识别和验证的生理特征或行为方式。生物特征分为身体特征和行为特征两类。身体特征包括：指纹、掌型、视网膜、虹膜、人体气味、脸型、手的血管和 DNA 等；行为特征包括：签名、语音、行走步态等。目前部分学者将视网膜识别、虹膜识别和指纹识别等归为高级生物识别技术；将掌型识别、脸型识别、语音识别和签名识别等归为次级生物识别技术；将血管纹理识别、人体气味识别、DNA 识别等归为“深奥的”生物识别技术，指纹识别技术目前应用广泛的领域有门禁系统、微型支付等。

### 2.9.7 双因素身份认证

所谓双因素就是将两种认证方法结合起来，进一步加强认证的安全性，目前使用最为广泛的双因素有：

- 动态口令牌 + 静态密码；
- USB Key + 静态密码；
- 二层静态密码等。

## 2.10 管理层：信息安全标准化组织及标准

### 2.10.1 国际信息安全标准概览

国际上信息安全标准化工作兴起于 20 世纪 70 年代中期，80 年代有了较快的发展，90 年代引起了世界各国的普遍关注。目前世界上有近 300 个国际和区域性组织制定标准或技术规则。与信息安全标准化有关的组织主要有以下 4 个。

- ISO（国际标准化组织）。ISO/IEC JTC1（信息技术标准化委员会）所属 SC27（安全技术分委员会）的前身是 SC20（数据加密技术分委员会），主要从事信息技术安全的一般方法和技术的标准化工作。而 ISO/TC68 负责银行业务应用范围内有关信息安全标准的制定，主要制定行业应用标准，与 SC27 有着密切的联系。ISO/IEC JTC1 负责制定的标准主要是开放系统互连、密钥管理、数字签名、安全评估等方面的内容。
- IEC（国际电工委员会）。IEC 在信息安全标准化方面除了与 ISO 联合成立了 JTC1 下分委员会外，还在电信、电子系统、信息技术和电磁兼容等方面成立技术委员会（如 TC56 可靠性、TC74 IT 设备安全和功效、TC77 电磁兼容、TC 108 音频/视频、信息技术和通信技术电子设备的安全等），并且制定相关国际标准（如信息技术设备安全 IEC60950 等）。
- ITU（国际电信联盟）。ITU SG17 组负责研究网络安全标准，包括通信安全项目、安全架构和框架、计算安全、安全管理、用于安全的生物测定、安全通信服务。此外 SG16

和下一代网络核心组也在通信安全、H.323 网络安全、下一代网络安全等标准方面进行研究。

- IETF（Internet 工程任务组）等。IETF 标准制定的具体工作由各个工作组承担。Internet 工程任务组分成 8 个工作组，分别负责 Internet 路由、传输、应用等 8 个领域，其著名的 IKE 和 IPSec 都在 RFC 系列之中，还有电子邮件、网络认证和密码及其他安全协议标准。

当前，国际上著名的信息安全评估标准有以下几种。

1）可信的计算机系统安全评估标准（TCSEC，从橘皮书到彩虹系列）

TCSEC 是由美国国防部于 1985 年公布的，是计算机系统信息安全评估的第一个正式标准。它把计算机系统的安全分为 4 类、7 个级别，对用户登录、授权管理、访问控制、审计跟踪、隐蔽通道分析、可信通道建立、安全检测、生命周期保障、文档写作、用户指南等内容提出了规范性要求。

2）信息技术安全评估标准（ITSEC，欧洲白皮书）

ITSEC 是由法、英、荷、德欧洲四国于 20 世纪 90 年代初联合发布的，它提出了信息安全的机密性、完整性、可用性的安全属性。机密性就是保证没有经过授权的用户、实体或进程无法窃取信息；完整性就是保证没有经过授权的用户不能改变或者删除信息，从而信息在传送的过程中不会被偶然或故意破坏，保持信息的完整、统一；可用性是指合法用户的正常请求能及时、正确、安全地得到服务或回应。ITSEC 把可信计算机的概念提高到可信信息技术的高度上来认识，对国际信息安全的研究、实施产生了深刻的影响。

3）信息技术安全评价的通用标准（CC）

CC 是由美、加、英、法、德、荷六国于 1996 年联合提出的，并逐渐形成国际标准 ISO15408。该标准定义了评价信息技术产品和系统安全性的基本准则，提出了目前国际上公认的表述信息技术安全性的结构，即把安全要求分为规范产品和系统安全行为的功能要求以及解决如何正确有效地实施这些功能的保证要求。CC 标准是第一个信息技术安全评价国际标准，它的发布对信息安全具有重要意义，是信息技术安全评价标准以及信息安全技术发展的一个重要里程碑。

4）ISO 13335 标准

ISO 13335 首次给出了关于 IT 安全的保密性、完整性、可用性、审计性、认证性、可靠性 6 个方面的含义，并提出了以风险为核心的安全模型：企业的资产面临很多威胁（包括来自内部的威胁和来自外部的威胁）；威胁利用信息系统存在的各种漏洞（如：物理环境、网络服务、主机系统、应用系统、相关人员、安全策略等），对信息系统进行渗透和攻击。如果渗透和攻击成功，将导致企业资产的暴露；资产的暴露（如系统高级管理人员由于不小心而导致重要机密信息的泄露），会对资产的价值产生影响（包括直接和间接的影响）；风险就是威胁利用漏洞使资产暴露而产生的影响的大小，这可以为资产的重要性和价值所决定；对企业信息系统安全风险的分析，便得出了系统的防护需求；根据防护需求的不同制定系统的安全解决方案，选择适当的防护措施，进而降低安全风险，并抗击威胁。该模型阐述了信息安全评估的思路，对企业的信息安全评估工作具有指导意义。

5）AS/NZS 4360:1999

AS/NZS 4360:1999是澳大利亚和新西兰联合开发的风险管理标准，第一版于1995年发布。在AS/NZS 4360:1999中，风险管理分为建立环境、风险识别、风险分析、风险评价、风险处置、风险监控与回顾、通信和咨询七个步骤。AS/NZS 4360:1999是风险管理的通用指南，它给出了一整套风险管理的流程，对信息安全风险评估具有指导作用。目前该标准已广泛应用于新南威尔士洲、澳大利亚政府、英联邦卫生组织等机构。

6）BS 7799

BS 7799是英国的工业、政府和商业共同需求而发展的一个标准，它分两部分：第一部分为“信息安全管理事务准则”；第二部分为“信息安全管理系统的规范”。目前此标准已经被很多国家采用，并已成为国际标准ISO 17799。BS 7799包含10个控制大项、36个控制目标和127个控制措施。BS 7799/ISO 17799主要提供了有效地实施信息系统风险管理的建议，并介绍了风险管理的方法和过程。企业可以参照该标准制定出自己的安全策略和风险评估实施步骤。

7）ISO/IEC 27001

信息安全管理实用规则ISO/IEC 27001的前身为英国的BS 7799标准，该标准由英国标准协会（BSI）于1995年2月提出，并于1995年5月修订而成。1999年BSI重新修改了该标准。BS 7799分为两个部分：BS 7799-1（信息安全管理实施规则）和BS 7799-2（信息安全管理体系规范）。第一部分对信息安全管理给出建议，供负责在其组织启动、实施或维护安全的人员使用；第二部分说明了建立、实施和文件化信息安全管理体系（ISMS）的要求，规定了根据独立组织的需要应实施安全控制的要求。2000年，国际标准化组织（ISO）在BS 7799-1的基础上制定并通过了ISO 17799标准。BS 7799-2在2002年也由BSI进行了重新修订。ISO组织在2005年对ISO 17799再次修订，BS 7799-2也于2005年被采用为ISO 27001:2005。

ISO/IEC 17799-2000（BS 7799-1）对信息安全管理给出建议，供负责在其组织启动、实施或维护安全的人员使用。该标准为开发组织的安全标准和有效的安全管理做法提供公共基础，并为组织之间的交往提供信任。

标准指出“像其他重要业务资产一样，信息也是一种资产”。它对一个组织具有价值，因此需要加以合适地保护。信息安全防止信息受到的各种威胁，以确保业务的连续性，使业务受到损害的风险减至最小，使投资回报和业务机会最大。

信息安全是通过实现一组合适控制获得的。控制可以是策略、惯例、规程、组织结构和软件功能。需要建立这些控制，以确保满足该组织的特定安全目标。

ISO/IEC 17799-2000包含了127个安全控制措施来帮助组织识别在运做过程中对信息安全有影响的元素，组织可以根据适用的法律法规和章程加以选择和使用，或者增加其他附加控制。国际标准化组织（ISO）在2005年对ISO 17799进行了修订，修订后的标准作为ISO 27000标准族的第一部分——ISO/IEC 27001，新标准去掉9点控制措施，新增17点控制措施，并重组部分控制措施而新增一章，重组部分控制措施、关联性逻辑性更好，更适合应用；并修改了部分控制措施措辞。修改后的标准包括11个章节。

- 安全策略
- 信息安全的组织
- 资产管理
- 人力资源安全

- 物理和环境安全
- 通信和操作管理
- 访问控制
- 系统采集、开发和维护
- 信息安全事故管理
- 业务连续性管理
- 符合性

ISO 27001:2005 系列标准主要内容包括以下几个组成部分。

- Chapter 0：简介 ISO 27001 控制目标和控制措施。
- Chapter 1：范围安全方针。
- Chapter 2：强制性应用标准安全组织。
- Chapter 3：术语和定义资产分类与控制。
- Chapter 4：信息安全管理体系人员安全、物理与环境安全、通信与运行管理系统开发与维护。
- Chapter 5：管理责任、访问控制。
- Chapter 6：SMS 内部审查信息安全事件管理。
- Chapter 7：ISMS 管理评审业务持续性管理。
- Chapter 8：ISMS 改善符合性。

### 2.10.2 国内信息安全标准概览

除国际之外，国内方面主要是等同采用国际标准。公安部主持制定、国家质量技术监督局发布的中华人民共和国国家标准 GB 17895—1999《计算机信息系统安全保护等级划分准则》已正式颁布并实施。该准则将信息系统安全分为 5 个等级：自主保护级、系统审计保护级、安全标记保护级、结构化保护级和访问验证保护级。主要的安全考核指标有身份认证、自主访问控制、数据完整性、审计等，这些指标涵盖了不同级别的安全要求。GB 18336 也是等同采用 ISO 15408 标准。

国内的安全标准组织主要有信息技术安全标准化技术委员会（CITS）以及中国通信标准化协会（CCSA）下辖的网络与信息安全技术工作委员会。CITS 成立于 1984 年，在国家标准化管理委员会和信息产业部的共同领导下负责全国信息技术领域以及与 ISO/IEC JTC1 相对应的标准化工作，目前下设 24 个分技术委员会和特别工作组，是国内最大的标准化技术委员会，也是具有广泛代表性、权威性和军民结合的信息安全标准化组织。CITS 主要负责信息安全的通用框架、方法、技术和机制的标准化及归口国内外对应的标准化工作，其中技术安全包括开放式安全体系结构、各种安全信息交换的语义规则、有关的应用程序接口和协议引用安全功能的接口等。CCSA 成立于 2002 年 12 月 18 日，是国内企事业单位自愿联合组织起来经业务主管部门批准的开展通信技术领域标准化活动的组织。CCSA 下设了有线网络信息安全、无线网络信息安全、安全管理和安全基础设施 4 个工作组负责研究：有线网络中电话网、互联网、传输网、接入网等在内所有电信网络相关的安全标准；无线网络中接入、核心网、业务等相关的安全标准以及安全管理工作组；安全基础设施工作组中网管安全以及安全基础设施相关的标准。

等级保护有以下三大项内容。

- 对信息和信息系统分等级进行保护，这也是一项核心工作，也是最重要的工作。

- 对信息系统安全专用产品分等级进行管理，将来各个单位使用的安全产品应该是分等级的。定了三级的系统不能使用二级以下的安全产品。43 号文对这方面已经做了一些初步的规定，但是详细的规定现在还没有正式出台。
- 对所发生的信息安全事件分等级进行响应和处置。

1）信息和信息系统分等级实行安全保护

我们国家把信息和信息系统按照重要程度，也就是对单位的利益、对社会公众的利益、对国家安全的影响三方面要素的重要程度来确定信息系统的安全等级。这个等级一共分为五级。第一级是最低的，第五级是最高的。信息安全等级保护制度是我们学习国外的先进方法，美国从 1983 年开始美国国防部出台了橘皮书，橘皮书把操作系统划分为四类八个等级。应该说，它是等级保护的先行者。当然这四类八个等级是根据技术程度所采用的安全机制多和少、强和弱来确定的。橘皮书名字是《可信计算机系统评测准则》。我们国家在 1994 年出台了国务院的 147 号令，《中华人民共和国计算机信息系统安全保护条例》里面已经明确规定，在我国要实行信息系统的等级保护制度。这就是等级保护的由来。这项工作是 1994 年提出，到 2004 年国务院的四大部委，公安部、国家保密局、国家密码局、前国务院信息化办公室，联合下了 66 号文，正式提出信息安全等级保护要在我国开始实施。当时，把信息安全等级划分了五个等级，起了相应的名字：

第一级：用户自主保护级

第二级：指导保护级

第三级：监督保护级

第四级：强制保护级

第五级：专控保护级

2）国家对信息安全产品的使用实行分等级管理

按照信息安全产品的使用要根据可控性、可靠性、安全性和可监督性这四个属性确定信息系统使用的安全产品等级。这四个属性在正式的文件里没有体现，43 号文里没有提到这四个属性。但是 43 号文里指出信息系统安全专用产品应该是国内生产的，源代码这些东西应该是国内生产的。43 号文里只是就安全性这个角度，给出了你使用安全产品必须是国产这样一个要求，但是还没有明确这个信息安全产品等级究竟应该怎么划分。

很多的安全厂商都说用户问到，我买了你的安全产品是不是达到了这个等级？你的安全产品是不是符合这样的等级？我给他们的回答是否定的。是否符合等级要看产品是否符合国家标准，如果符合了标准，那就符合安全等级；如果没有符合标准那就不符合安全等级。第三级从技术的角度应该对应着标记保护级，标记保护级里面应该要求实行最小授权，不应该再有超级管理员的身份。超级管理员必须分别为系统管理员、系统安全员和审计员。权限低的人的活动都可以查询。但是管理员本人做的坏事，要把这个痕迹都擦掉，这个问题该怎么解决？对于三级以上的信息系统，我们就要求这些东西是不能擦掉的，至少审计员必须是独立出来的。目前我们的安全产品虽然也设了审计员，但是超级用户还是存在的，这个问题就没有达到标记保护级的要求。这是一个漫长的过程，昨天郭处长也在讲，从现在推动，通过检查、测评、整改，真正能达到初步规范，他的预计是三年时间，我考虑对各大部委来说三年的时间可能会实现，就全国来说，三年肯定是不够的。而且我们的安全产品提升到新的水平也需要过程。

3）信息安全事件实行分等级响应处置

国家已经把信息安全事件以国标的形式进行了分类和分等级。这些分类和分等级依据国标，对不同的信息安全事件，由监管部门牵头组织全社会的应急响应和本单位的应急响应相结合，最大限度地减轻信息安全事件造成的损失。

等级保护的法律依据及技术标准：人民警察法，计算机信息系统安全保护条例，43 号文，信息安全等级保护管理办法，等级保护的各类文件，中办、国办的 27 号文，04 年 66 号文，06 年 7 号文，07 年 43 号文。

等级保护技术标准：从技术的角度讲必须依据国家相应的安全标准。实际上不仅仅是在技术方面，在管理方面，工程方面都应该按照国家的标准做。我们国家出台的第一个关于等级保护的标准是 1999 年出台的国标 17859，这个标准叫做《计算机信息系统安全等级划分准则》。这里面把信息系统按照安全机制的多与少，强与弱也划分了五个等级。

第一级：用户自主保护级

第二级：系统审计保护级

第三级：安全标记保护级

第四级：结构化保护级

第五级：访问验证保护级

这五个等级和前面说的 1、2、3、4、5 有相同的地方，又有不同的地方。前面的五个等级怎么确定，并不是根据采取的安全机制的多少和强弱来确定等级，而是根据信息资产的重要程度来确定安全等级。

安全等级的确定基本依据是数据的重要程度和系统的服务功能的重要程度，由这两个重要程度决定信息资产应该具备的等级。国家已经出台了一个关于定级的指南——信息系统安全保护定级指南，大家可以看一下。而且定级的工作，已经开始，第一期的定级工作也已经在全国结束了，当然以后还会有一些单位接着定级，来做这些工作。第一期最重要的部分——单位定级已经问世，并且到公安部门已经备案。

17859 等级划分准则由于是 1999 年出台的，很多问题没有考虑到，这个标准实际等于是橘皮书的一个翻版。所以对一些问题考虑得还不够完善。橘皮书主要是从信息的机密性角度给出了四类八个等级：D 类（没有任何保护的）、C 类（分为 C1 和 C2 级）、B 类（分为 B1、B2、B3）、A 类和超 A 类。

超 A 类没有描述，A 类有描述但是实现起来还有很多问题，尤其是在我国。我国制定这些标准的人，把橘皮书去了头和尾，把 D 类、A 类和超 A 类去掉，只留下 B 类和 C 类，就出现了前面说的这五个等级。第一级和第二级是 C1 级和 C2 级，第三级、第四级、第五级对应的是 B 类的 B1、B2、B3。我们比橘皮书有一点很重要的进步就是提到了完整性保护的问题。虽然提了，但是没有细致地讲这个完整性应该保护到什么程度，怎么保护。

由于 17859 的这些不足，2002 年公安部又组织了很多专家起草了公共行业安全标准——387—392。这个标准对 17859 做了一个很好的说明和补充。这条标准到 2006 年，被提升为国家标准，即 GB/T 20269—20273 标准。

第十二条在信息系统建设过程中，运营、使用单位应当按照《计算机信息系统安全保护等级划分准则》、《信息系统安全等级保护基本要求》等技术标准，参照之前的一些标准做一些等级保护措施。第十三条提出应该按照这些标准建设安全管理制度。

这是已经出台的国家等级保护体系，有一些产品的标准，事件处理的标准，服务的标准。这些标准有的已经作为国标正式发布，相当一部分属于报批稿，没有完全正式发布。

等级保护的开展：对于一个单位来说就是五大步骤，系统的定级、备案，这一个阶段就全国来说最重要的部分已经完成。接下来的工作是要做安全规划和设计，根据安全规划设计的结果做安全实施，然后是一个长期的运行维护过程，最后是系统的废止。

# 第二篇

# 企业 Linux 安全运维规划及选型

## 凡事预则立，不预则废

《礼记·中庸》一书中有这样一句话：“凡事预则立，不预则废。”意思是说，要想成就任何一件事，必须要有明确的目标，认真的准备和周密的安排。没有准备的盲目行动，只能是虽忙忙碌碌却一事无成。预，就是准备，是努力，是奋斗，是实践，是付出；立，则是成功。有了精心的准备，艰苦的努力，不懈的奋斗，扎实的实践和巨大的付出，才能达到成功的彼岸。所以说，预是成功的基础，不预则是失败的根源。

企业安全建设和运维工作也是如此，需要从软硬件选型、信息安全建设的范围和内容等方面进行整体的规划。

# 第3章

# 规划：企业信息安全工作思路

## 本章导读

前 2 章介绍了当前信息安全面临的现状、威胁和业界常用的一些技术，使企业的 CTO、CIO 等能够对它们有充分的认识和警惕，所谓“知己知彼，百战不殆”。但是，很多人都会有这么一个疑问，作为 500 强的企业来说，企业信息安全工作如何展开呢？具体到企业 Linux 安全的规划和运维又包括哪些方面呢？前面说的难免有些虚无缥缈，感觉不是非常详细和具有指导性，那么笔者将在本章来介绍一下该 500 强企业的信息安全工作的工作思路，以及具体的 Linux 安全的实施内容，解除大家的疑惑。

## 3.1 信息安全的本质

“信息安全”曾经仅是学术界所关心的术语，就像是五、六十年前“计算机”被称为“电算机”那样仅被学术界所了解一样。现在，“信息安全”因各种原因已经像公众词汇那样被广大公众所熟知，尽管尚不能与“计算机”这个词汇的知名度所比拟，但也已经具有广泛的普及性。问题的关键在于人们对“计算机”的理解不会有什么太大的偏差，而对“信息安全”的理解则往往各式各样。种种偏差主要来自于从不同的角度来看信息安全，因此出现了“计算机安全”、“网络安全”、“信息内容安全”之类的提法，也出现了“机密性”、“真实性”、“完整性”、“可用性”、“不可否认性”等描述方式。

关于信息安全的定义，以下是一些有代表性的定义方式。

- 国内学者给出的定义是：“信息安全保密内容分为：实体安全、运行安全、数据安全和管理安全四个方面。”
- 我国计算机信息系统安全专用产品分类原则给出的定义是：“涉及实体安全、 运行安全和信息安全三个方面。”
- 我国相关立法给出的定义是：“保障计算机及其相关的和配套的设备、设施（网络）的安全，运行环境的安全，保障信息安全，保障计算机功能的正常发挥，以维护计算机信息系统的安全”。这里面涉及了物理安全、运行安全与信息安全三个层面。
- 国家信息安全重点实验室给出的定义是：“信息安全涉及到信息的机密性、完整性、可用性、可控性。综合起来说，就是要保障电子信息的有效性。”
- 英国 BS7799 信息安全管理标准给出的定义是：“信息安全是使信息避免一系列威胁，保障商务的连续性，最大限度地减少商务的损失，最大限度地获取投资和商务的回报，涉及的是机密性、完整性、可用性。”
- 美国国家安全局信息保障主任给出的定义是：“因为术语‘信息安全’一直仅表示信息的机密性，在国防部我们用‘信息保障’来描述信息安全，也叫‘IA’。它包含 5 种安全服务，包括机密性、完整性、可用性、真实性和不可抵赖性。”
- 国际标准化委员会给出的定义是：“为数据处理系统而采取的技术的和管理的安全保护，保护计算机硬件、软件、数据不因偶然的或恶意的原因而遭到破坏、更改、泄露”。这里面既包含了层面的概念，其中计算机硬件可以看作是物理层面，软件可以看作是运行层面，再就是数据层面；又包含了属性的概念，其中破坏涉及的是可用性，更改涉及的是完整性，显露涉及的是机密性。

纵观从不同的角度对信息安全的不同描述，可以看出两种描述风格：一种是从信息安全所涉及的层面的角度进行描述，大体上涉及了实体（物理）安全、运行安全、数据（信息）安全；一种是从信息安全所涉及的安全属性的角度进行描述，大体上涉及了机密性、完整性、可用性。

信息安全出现多种不同说法，存在着多种观察视角，并不是偶然的现象。信息安全的发展历史、信息安全的作用层面、信息安全的基本属性，都决定了信息安全的不同内涵。

从信息安全的发展历史来看，在二十世纪四、五十年代，人们认为信息安全就是通信保密，采用的保障措施就是加密和基于计算机规则的访问控制，这个时期被称为“通信保密（COMSEC）”时代，其时代标志是 1949 年 Shannon 发表的《保密通信的信息理论》；在七十年代，人们关心

的是计算机系统不被非授权的他人使用，这时学术界称之为“计算机安全（INFOSEC）”时代，其时代特色是美国八十年代初发布的橘皮书——可信计算机评估准则（TCSEC）；九十年代，人们关心的是如何防止通过网络对联网计算机进行攻击，这时学术界称之为“网络安全（NETSEC）”，其时代特征是美国八十年代末出现的“莫里斯”蠕虫事件；进入了二十一世纪，人们关心的是信息及信息系统的保障，如何建立完整的保障体系，以便保障信息及信息系统的正常运行，这时学术界称之为“信息保障（IA）”。

从信息安全的作用层面来看，人们首先关心的是计算机与网络的设备硬件自身的安全，就是信息系统硬件的稳定性运行状态，因而称之为“物理安全”；其次人们关心的是计算机与网络设备运行过程中的系统安全，就是信息系统软件的稳定性运行状态，因而称之为“运行安全”；当讨论信息自身的安全问题时，涉及的就是狭义的“信息安全”问题，包括对信息系统中所加工存储、网络中所传递的数据的泄露、仿冒、篡改以及抵赖过程所涉及的安全问题，称之为“数据安全”。因此，就信息安全作用点来看，可以称之为信息安全的层次模型，这也是国内学者普遍认同的定义方式，如图 3-1 所示。

| 数据（信息）安全 |
|---|
| 运行（系统）安全 |
| 物理（实体）安全 |

图 3-1　一种信息安全的层次模型

从信息安全的基本属性来看，机密性就是对抗对手的被动攻击，保证信息不泄露给未经授权的人，或者即便数据被截获，其所表达的信息也不被非授权者所理解。完整性就是对抗对手主动攻击，防止信息被未经授权者篡改。可用性就是确保信息及信息系统能够为授权使用者所正常使用。这三个重要的基本属性被国外学者称为“信息安全金三角”（CIA，Confidentiality-Integrity-Avaliability），如图 3-2 所示。

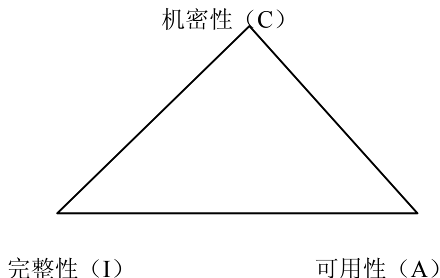



图 3-2　信息安全金三角模型

## 3.2　信息安全概念经纬线：从层次到属性

从信息安全的作用层次来看，前面已经介绍了人们所关注的三个层面，即物理安全层、运行安全层以及数据安全层。但是，还有两个层面尚没有人在同一个框架之下给出清晰的描述。一个是关于信息内容的安全问题，一个是关于信息对抗的问题，而这两个层面的安全问题也是业界普遍关心的问题。所不同的是，内容安全更被文化、宣传界人士所关注；而信息对抗则更被电子对抗研究领域的人士所关注。

信息内容安全的问题已经深刻地展现在现实社会的面前，主要表现在有害信息利用互联网所提供的自由流动的环境肆意扩散，其信息内容或者像脚本病毒那样给接收的信息系统带来破坏性的后果，或者像垃圾邮件那样给人们带来烦恼，或者像谣言那样给社会大众带来困惑，从而成为社会不稳定因素。但是，就技术层面而言，信息内容安全技术的表现形式是对信息流动的选择控制能力，换句话说，表现出来的是对数据流动的攻击特性。

信息对抗严格地说是信息谋略范畴的内容，是讨论如何从多个角度或侧面来获得信息并分析

信息，或者在信息无法隐蔽的前提下，通过增加更多的无用信息来扰乱获取者的视线，以掩藏真实信息所反映的含义。从本质上来看，信息对抗是在信息熵的保护或打击层面上讨论问题，也就是围绕着信息的利用来进行对抗。

从信息安全的属性来看，除了前面所介绍的机密性、完整性、可用性等三个基本属性之外，还有更多的一些属性也用于描述信息安全的不同特性，如真实性、可控性、合法性、实用性、占有性、唯一性、不可否认性、可追溯性、生存性、稳定性、可靠性、特殊性等。其中，真实性反映的是主体身份、行为及相关信息的真实有效；可控性反映的是信息系统不会被非授权使用，信息的流动可以被选择性阻断；合法性反映的是信息或信息系统的行为获得了授权；实用性反映的是信息加密密钥与信息的强关联性及密钥不可丢失的特性；占有性反映的是信息载体不可被盗用，确保由合法信息所占有；唯一性反映的是各信息以及信息系统行为主体之间不会出现混淆；不可否认性反映的是所生成的信息或信息系统的合法行为不会被抵赖；可追溯性反映的是信息系统的行为具有被审计的能力；生存性反映的是信息系统在受到攻击时可以通过采取降级等措施以保持最低限度的核心服务能力；稳定性反映的是信息系统不会出现异常情况；可靠性反映的是信息系统能够保持正常运行而不受外界影响的能力；特殊性反映的是信息所需要表现的特定的内涵。

基于信息安全的作用点，从信息系统、信息以及信息熵的角度，可以将信息安全看作是由三个层次、五个层面所构成的层次框架体系，并且在每个层面中各自反映了在该层面中所涉及的信息安全的属性，如表 3-1 所示。

**表 3-1　信息安全的层次框架体系**

| 层　次 | 层　面 | 作　用　点 | 安全属性 |
|---|---|---|---|
| 信息对抗 | 信息熵对抗 | 信息利用 | 机密性、完整性、特殊性 |
| 信息安全 | 内容安全 | 攻击信息 | 机密性、真实性、可控性、可用性、完整性、可靠性 |
|  | 数据安全 | 保护信息 | 机密性、真实性、实用性、完整性、唯一性、不可否认性、生存性 |
| 系统安全 | 运行安全 | 软件 | 真实性、可控性、可用性、合法性、唯一性、可追溯性、占有性、生存性、稳定性、可靠性 |
|  | 物理安全 | 硬件 | 机密性、可用性、完整性、生存性、稳定性、可靠性 |

在此，系统安全反映的是信息系统所面临的安全问题。其中，物理安全涉及的是硬件设施方面的安全问题；运行安全涉及的是操作系统、数据库、应用系统等软件方面的安全问题。狭义的信息安全所反映的是信息自身所面临的安全问题。其中，数据安全是以保护数据不受外界的侵扰为目的，包括与泄密、伪造、篡改、抵赖等有关的行为；内容安全则是反过来对流动的数据进行限制，包括可以对指定的数据进行选择性地阻断、修改、转发等特定的行为。信息对抗则是针对信息中的信息熵而进行的隐藏、掩盖，或发现、分析的行为。

从信息安全属性的角度来看，每个层面所涉及的信息安全的属性，对应于该层面信息安全的外部特征，也具有相应的处置方法。

（1）物理安全。是指对网络与信息系统的物理装备的保护，主要涉及网络与信息系统的机密性、可用性、完整性、生存性、稳定性、可靠性等基本属性。所面对的威胁主要包括电磁泄漏、通信干扰、信号注入、人为破坏、自然灾害、设备故障等。主要的保护方式有加扰处理、电磁屏蔽、数据校验、容错、冗余、系统备份等。

（2）运行安全。是指对网络与信息系统的运行过程和运行状态的保护，主要涉及网络与信息

系统的真实性、可控性、可用性、合法性、唯一性、可追溯性、占有性、生存性、稳定性、可靠性等。所面对的威胁包括非法使用资源、系统安全漏洞利用、网络阻塞、网络病毒、越权访问、非法控制系统、黑客攻击、拒绝服务攻击、软件质量差、系统崩溃等。主要的保护方式有防火墙与物理隔离、风险分析与漏洞扫描、应急响应、病毒防治、访问控制、安全审计、入侵检测、源路由过滤、降级使用、数据备份等。

（3）数据安全。是指对信息在数据收集、处理、存储、检索、传输、交换、显示、扩散等过程中的保护，使得在数据处理层面保障信息依据授权使用，不被非法冒充、窃取、篡改、抵赖，主要涉及信息的机密性、真实性、实用性、完整性、唯一性、不可否认性、生存性等；所面对的威胁包括窃取、伪造、密钥截获、篡改、冒充、抵赖、攻击密钥等。主要的保护方式有加密、认证、非对称密钥、完整性验证、鉴别、数字签名、秘密共享等。

（4）内容安全。是指对信息在网络内流动中的选择性阻断，以保证信息流动的可控能力。在此，被阻断的对象可以是通过内容可以判断出来的对系统造成威胁的脚本病毒、因无限制扩散而导致消耗用户资源的垃圾类邮件、导致社会不稳定的有害信息，等等。主要涉及信息的机密性、真实性、可控性、可用性、完整性、可靠性等。所面对的难题包括信息不可识别（因加密）、信息不可更改、信息不可阻断、信息不可替换、信息不可选择、系统不可控等。主要的处置手段是密文解析或形态解析、流动信息的裁剪、信息的阻断、信息的替换、信息的过滤、系统的控制等。

（5）信息对抗。是指在信息的利用过程中，对信息熵的真实性的隐藏与保护，或者攻击与分析。主要涉及信息熵的机密性、完整性、特殊性等。所面对的主要问题包括多角度综合分析、攻击或压制信息的传递、用无用信息来干扰信息熵的本质。主要的处置手段是消隐重要的局部信息、加大信息获取能力、消除信息的不确定性等。

## 3.3 业界信息安全专家定义的信息安全：信息安全四要素

业界著名的信息安全专家方滨兴院士在深入分析和继承了传统信息安全的定义前提下，根据当前国际信息安全的发展现状，给出了信息安全四要素，并重新概括和界定了信息安全的内涵和外延。

在诸多信息安全的属性中，分析可见，机密性、真实性、可控性、可用性属于基本属性，相互不能蕴涵。其中机密性反映了信息与信息系统的不可被非授权者所利用；真实性反映了信息与信息系统的行为不被伪造、篡改、冒充；可控性反映了信息的流动与信息系统可被控制者所监控；可用性反映了信息与信息系统可被授权者所正常使用。而其他属性，包括实用性、完整性、合法性、唯一性、不可否认性、特殊性、占有性、可追溯性、生存性、稳定性、可靠性等，则属于上述四个基本属性的某个侧面的突出反映，因此可以归结为这四个基本属性之中。其中实用性反映的是机密性在密钥依赖方面的机密属性；完整性、合法性、唯一性、不可否认性、特殊性分别反映的是真实性在信息内容本身、信息来源、信息系统行为主体、信息的发布行为、信息熵方面的真实属性；占有性、可追溯性分别反映的是可控性对信息资源的保护、对信息及信息系统行为的审计能力；生存性、稳定性、可靠性分别反映的是可用性在信息系统的容灾能力、信息系统的健壮能力、信息系统的可靠能力方面的可用属性。由此，机密性、真实性、可控性、可用性这四个基本属性实际上就是信息安全的四个核心属性，可以反映出信息安全的基本概貌。相对信息安全

金三角而言，可称之为信息安全四要素，简称CACA，如图3-3所示。

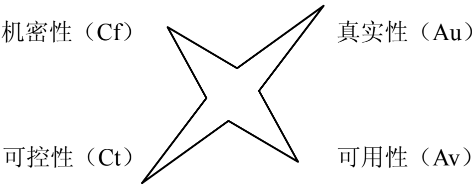



图3-3　信息安全四要素

与信息安全金三角相比，信息安全四要素弥补了金三角的明显不足。在金三角CIA的机密性、完整性、可用性三个属性中，完整性本来是一个小概念，是指确保信息内容不被篡改，而信息的来源甚至信息发布行为主体是否真实则无法表现。当然随着研究的深入，英国的BS7799信息安全管理标准中将完整性的概念扩大成涵盖了信息发布者的真实与信息来源的真实两个，但这明显是牵强地将固有的概念术语的外延扩大到本不属于这一术语的意义上。反过来，真实性则完全可以涵盖完整性，因为真实性不仅可以反映信息来源的真实、信息发布主体的真实，而且也可以反映信息内容的真实，而后者恰好是完整性的概念。另外，可控性描述了内容安全中对网络上的信息流具有可控能力的属性，而这在机密性、完整性与可用性中都无法表现。因而说，CIA模型是有着缺陷的。

假定在信息系统、信息与信息的利用中存在一种“序”，并且这种“序”反映了信息系统、信息与信息利用的正常运行状态，则针对信息系统、信息与信息利用的“序”的保持与攻击则可以看作是信息安全的问题。根据这一思路，我们重新定义信息安全的概念如下。

信息安全是对信息系统、信息与信息利用的固有属性（即“序”）的攻击与保护的过程。它围绕着信息系统、信息及信息熵的机密性、真实性、可控性、可用性这四个核心安全属性，具体反映在物理安全、运行安全、数据安全、内容安全和信息对抗五个层面上。

综合信息安全的层次特性与安全属性特性，可以形成一个信息安全概念的经纬线，如表3-2所示。

表3-2　信息安全概念的经纬线

| | 机 密 性 | 真 实 性 | 可 控 性 | 可 用 性 |
|---|---|---|---|---|
| 物理安全 | √ | √ | | √ |
| 运行安全 | | √ | √ | √ |
| 数据安全 | √ | √ | | √ |
| 内容安全 | √ | √ | √ | √ |
| 信息对抗 | √ | √ | | |

## 3.4 企业信息安全的实施内容和依据（框架）

### 3.4.1 基本原则

根据上述安全防范架构的多个层面的问题，综合考虑可实施性、可管理性、可扩展性、综合完备性、系统均衡性等方面，企业信息安全防范体系在整体设计过程中需要遵循以下几项准则，以切实全面地做好企业安全防范工作。

1．做好整体规划

企业信息安全系统应该包括安全防护机制、安全检测机制、安全响应机制和安全策略，这主要来源于经典的信息安全领域的 P2DR 模型（见图 3-4）。P2DR 模型包含四个主要部分：Policy（安全策略）、Protection（防护）、Detection（检测）和 Response（响应）。

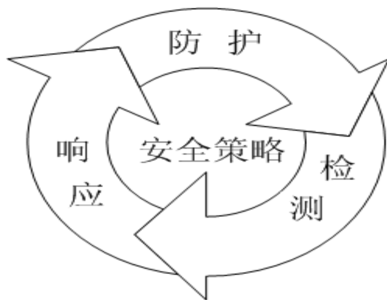


图 3-4 经典信息安全 P2DR 模型

- Policy（安全策略）。由于安全策略是安全管理的核心，所以要想实施动态网络安全循环过程，必须首先制定企业的安全策略，所有的防护、检测、响应都是依据安全策略实施的，企业安全策略为安全管理提供管理方向和支持手段。对于一个策略体系的建立包括：安全策略的制订、安全策略的评估、安全策略的执行等。
- Protection（防护）。防护通常是采用一些传统的静态安全技术及方法来实现的，主要有防火墙、加密、认证等方法。通过防火墙监视限制进出网络的数据包，可以防范外对内及内对外的非法访问，提高了网络的防护能力，当然需要根据安全策略制定合理的防火墙策略；也可以利用 SecureID 这种一次性口令的方法来增加系统的安全性等。
- Detection（检测）。在网络安全循环过程中，检测是非常重要的一个环节，检测是动态响应的依据，它也是强制落实安全策略的有力工具，通过不断地检测和监控网络和系统，来发现新的威胁和弱点，通过循环反馈来及时作出有效的响应。
- Response（响应）。紧急响应在安全系统中占有最重要的地位，是解决安全潜在性最有效的办法。从某种意义上讲，安全问题就是要解决紧急响应和异常处理问题。要解决好紧急响应问题，就要制订好紧急响应的方案，做好紧急响应方案中的一切准备工作。

因此，应用到企业安全防范架构上来说，我们也要很好地考虑这些要点，而不能为了防范而防范，要整体规划和部署。也就是说，安全防护机制是根据具体系统存在的各种安全威胁采取的相应的防护措施，避免非法攻击的进行。安全检测机制是检测系统的运行情况，及时发现和制止对系统进行的各种攻击。安全响应机制是在安全防护机制失效的情况下，进行应急处理和尽量、及时地恢复信息，减少供给的破坏程度。

2．做好层次性防范

层次性防范是指安全层次和安全级别。良好的信息安全系统必然是分为不同等级的，包括对信息保密程度的分级、对用户操作权限的分级、对网络安全程度的分级（安全子网和安全区域）、对系统实现结构的分级（应用层、网络层、链路层等），从而针对不同级别的安全对象，提供全面、可选的安全算法和安全体制，以满足网络中不同层次的各种实际需求。

3．突出重点，合理平衡

网络信息安全的木桶原则是指对信息均衡、全面地进行保护。网络信息系统是一个复杂的计算机系统，它本身在物理上、操作上和管理上的种种漏洞构成了系统的安全脆弱性，尤其是多用户网络系统自身的复杂性、资源共享性使单纯的技术保护防不胜防。攻击者使用的“最易渗透原则”，必然在系统中最薄弱的地方进行攻击。因此，充分、全面、完整地对系统的安全漏洞和安全威胁进行分析、评估和检测（包括模拟攻击）是设计信息安全系统的必要前提条件，并且，突

出一些重点环节。安全机制和安全服务设计首先应考虑到防止最常用的攻击手段，根本目的是提高整个系统的“安全最低点”的安全性能，从而保证既做到合理的平衡，又做到了重点突出。

4．技术与管理，两手都要硬

安全体系是一个复杂的系统工程，涉及人、技术、操作等要素，单靠技术或单靠管理都不可能实现。因此，必须将各种安全技术与运行管理机制、人员思想教育与技术培训、安全规章制度建设相结合。切忌只重视其中一种而忽视另一种的情况出现，要明白好的技术是管理的基础，而高效地管理则是技术强有力的保证。

5．模块化，可动态调整

当今网络时代的技术日新月异，网络攻与防的较量也在不断地升级，新的黑客技术和网络威胁的出现，必然要求我们要有新的手段和对策进行防御。所以我们的安全防护不可能一步到位，可在一个比较全面的安全规划下，根据网络的实际需要，先建立基本的安全体系，保证基本的、必须的安全性。今后随着网络规模的扩大及应用的增加、网络应用和复杂程度的变化，网络脆弱性也会不断增加，调整或增强安全防护力度，保证整个网络最根本的安全需求。要根据网络安全的变化不断调整安全措施，适应新的网络环境，满足新的网络安全需求。另外，各个网络防御手段和技术需要考虑到模块化的原则，尽量做到相互之间不要有太大的依赖性，否则很容易导致单点失效的问题，对它们进行合理的模块化，既能保证较好的防护效果，又能达到可动态调整和扩展的目的。

## 3.4.2 传统的企业信息安全架构

从当前的理论研究以及实践经验来看，面向企业安全的防范体系（架构）并没有一个成型的模型，各企业均按照自身的经验和组织管理体系来进行企业网络安全的防范工作。然而，在实践中急需有一套具有指导性意义的方法和手段来对其工作进行指导，才能做到有备无患。我们看到，关于网络安全的相关标准及模型的研究已经日趋成熟和理论化，并在实践中广泛应用。因此，传统的做法是借用这些模型和标准来构建一套较为有效和具有普遍意义的企业安全防范体系。ITU-T X.800 Security Architecture标准将我们常说的“网络安全（network security）”进行逻辑上的分别定义，即安全攻击（security attack）是指损害机构所拥有信息安全的任何行为；安全机制（security mechanism）是指设计用于检测、预防安全攻击或者恢复系统的机制；安全服务（security service）是指采用一种或多种安全机制以抵御安全攻击、提高机构的数据处理系统安全和信息传输安全的服务。

为了能够有效了解用户的安全需求，选择各种安全产品和策略，有必要建立一些系统的方法来进行网络安全防范。网络安全防范体系的科学性、可行性是其可顺利实施的保障。图3-5给出了DISSP安全框架三维模型。第一维是安全服务，

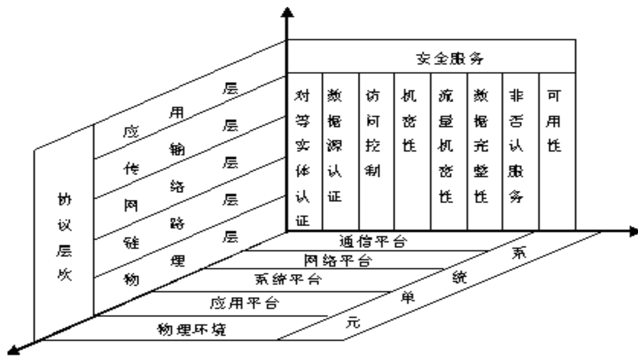


图3-5 DISSP安全框架三维模型

给出了八种安全属性。第二维是系统单元，给出了信息网络系统的组成。第三维是结构层次，给出并扩展了国际标准化组织 ISO 的七层开放系统互联（OSI）模型。

框架结构中的每一个系统单元都对应于某一个协议层次，需要采取若干种安全服务才能保证该系统单元的安全。网络平台需要有网络结点之间的认证、访问控制，应用平台需要有针对用户的认证、访问控制，需要保证数据传输的完整性、保密性，需要有抗抵赖和审计的功能，需要保证应用系统的可用性和可靠性。针对一个信息网络系统，如果在各个系统单元都有相应的安全措施来满足其安全需求，则我们认为该信息网络是安全的。

### 3.4.3　新的企业信息安全框架及其实施内涵

从上一小节的介绍不难看出，传统的企业信息安全框架存在以下几个问题。

- 概念含糊不清，且没有给出实施的可用参考。只是列出了信息安全需要关注的协议层面、系统单元和涉及的几个安全属性；而在安全属性中，流量机密性和机密性本来就是一回事，所采用的技术也是大同小异，并没有给出企业在信息安全保障实施过程中的任何可行性建议和手段；
- 忽略了管理安全。该框架只强调技术，而忽略了管理在企业信息安全保障中的重要作用和地位。所谓“三分技术，七分管理”，没有管理的技术难以落到实处，缺乏管理的指导性，盲目地使用技术也是不合理的。

所以，为了根据 500 强企业的自身特点来制定可行的企业信息安全框架，我们可以回顾一下方滨兴院士提出的信息安全定义。方滨兴院士提出的信息安全新的定义和内涵有普遍的适用性和实践指导意义，它尤其适用于国家信息安全工作的指导和规范。而我们 500 强企业在参照该定义的前提下，结合企业在信息安全工作的特点，将其中的“信息对抗”改进为“管理安全”，这主要是因为以下两个重要原因。

- 企业的信息安全工作主要是“防”，以防为主，立足自身，基本上不会采取信息对抗的方式来还击外部黑客和不法用户。
- 企业的信息安全工作很大一部分在于满足外部对企业的审核要求，企业对自身员工、资源等的管理要求，这就依赖于管理安全，他们需要参考和遵循许多业界成熟的标准和制度，比如 ISO/IEC 27001、萨班斯法案等。

因此，我们形成了企业信息安全框架。其本质是：企业信息安全从技术角度来看是对信息与信息系统固有属性的攻击与保护的过程。它围绕着信息系统、信息自身及信息利用的机密性、真实性、完整性、可控性、可用性、不可抵赖性这六个核心安全属性，具体反映在物理安全、运行安全、数据安全、内容安全、管理安全五个层面上，如图 3-6 所示。

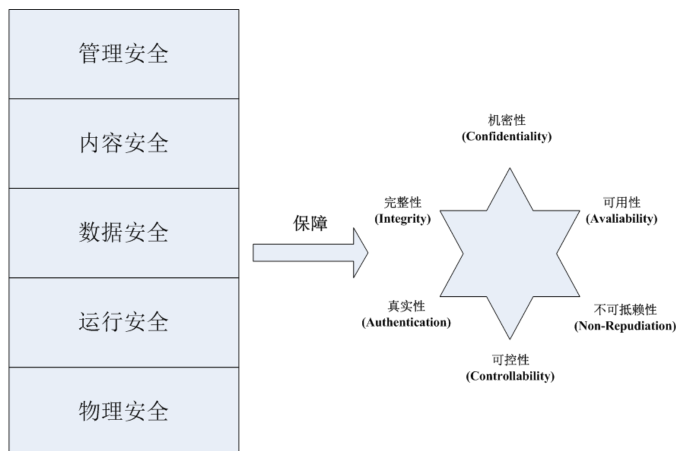



图 3-6　企业信息安全框架

根据图 3-6 的企业信息安全框架，在实践的过程中，需要采用各种各样的技术来完成多个安全任务，并参照相应的业界成熟的法规和制度来进行检查，从而完成企业信息安全工作，具体内容如表 3-3 所示。

表 3-3 企业信息安全管理详表

| 安全层面 | 含 义 | 安全任务列表 | 安全管理依据的相关制度和法规 |
| --- | --- | --- | --- |
| 物理安全 | 指对网络与信息系统物理装备的保护。主要涉及网络与信息系统的机密性、可用性、完整性等属性 | （1）干扰处理、电磁屏蔽：防范电磁泄漏；<br>（2）容错、容灾、冗余备份、生存性技术：防范随机性故障；<br>（3)信息验证：防范信号插入；<br>（4）机房管理：防范非法用户接触和破坏物理设备 | （1）GB 50174－93《电子计算机机房设计规范》；<br>（2）GA/T 390－2002《计算机信息系统安全等级保护通用技术要求》；<br>（3）GB 2887－2000《电子计算机场地通用规范》；<br>（4）GB 9361－88《计算站场地安全要求》 |
| 运行安全 | 指对网络与信息系统的运行过程和运行状态的保护。主要涉及网络与信息系统的真实性、可控性、可用性等 | （1）建立风险评估体系、安全测评体系：支持系统评估；<br>（2）部署漏洞扫描、采用安全协议：支持对安全策略的评估与保障；<br>（3）实施防火墙、物理隔离系统、访问控制技术、防恶意代码技术：支持访问控制；<br>（4）建立入侵检测、入侵防护及预警系统、部署安全审计技术：支持入侵检测和防护；<br>（5）采用反制系统、容侵技术、审计与追踪技术、取证技术：支持应急响应；<br>（6）采用防网络攻击技术，包括 Phishing、Botnet、DDoS、木马、社会工程学等防护技术；<br>（7）采用可信计算技术、安全操作系统技术、安全数据库技术：保证基础平台运行安全；<br>（8）采用 VLAN、网段隔离技术：支持细粒度的安全控制和策略；<br>（9）采用数据备份和灾难恢复技术：支持重要数据的备份和在灾难情况下的恢复 | （1）GA/T 681－2007 信息安全技术网关安全技术要求；<br>（2）GA/T 682－2007 信息安全技术路由器安全技术要求；<br>（3）GA/T 683－2007 信息安全技术防火墙安全技术要求；<br>（4）GA/T 684－2007 信息安全技术交换机安全技术要求；<br>（5）GA/T 685－2007 信息安全技术交换机安全评估准则；<br>（6）GA/T 697－2007 信息安全技术静态网页恢复产品安全功能要求；<br>（7）GA/T 698－2007 信息安全技术信息过滤产品安全功能要求；<br>（8）GA/T 699－2007 信息安全技术计算机网络入侵报警通信交换技术要求；<br>（9）GA/T 700－2007 信息安全技术计算机网络入侵分级要求；<br>（10)GB/T 20008－2005 信息安全技术操作系统安全评估准则；<br>（11)GB/T 20275－2006 信息安全技术入侵检测系统技术要求和测试评价方法；<br>（12）GB/T 20273－2006 信息安全技术 数据库管理系统安全技术要求；<br>（13)GB/T 20278－2006 信息安全技术 网络脆弱性扫描产品技术要求 |

续表

| 安全层面 | 含　义 | 安全任务列表 | 安全管理依据的相关制度和法规 |
|---|---|---|---|
| 数据安全 | 指对信息在数据收集、处理、存储、检索、传输、交换、显示、扩散等过程中的保护，使得在数据处理层面保障信息依据授权使用，不被非法冒充、窃取、篡改、抵赖。主要涉及信息的机密性、真实性、完整性、不可抵赖性等 | （1）采用对称与非对称密码技术及其硬化技术、VPN 等技术：防范信息泄密；<br>（2）采用认证、鉴别、PKI 等技术：防范信息伪造；<br>（3）采用完整性验证技术：防范信息篡改；<br>（4）采用数字签名技术：防范信息抵赖；<br>（5）采用秘密共享技术：防范信息破坏；<br>（6）采用隐写技术、水印技术：保护信息 | （1）GB/T 20518－2006 信息安全技术公钥基础设施数字证书格式；<br>（2）GB/T 20519－2006 信息安全技术公钥基础设施特定权限管理中心技术规范；<br>（3）GB/T 20520－2006 信息安全技术公钥基础设施时间戳规范；<br>（4）GB/T 21053－2007 信息安全技术公钥基础设施 PKI 系统安全等级保护技术要求；<br>（5）GB/T 21054－2007 信息安全技术公钥基础设施 PKI 系统安全等级保护评估准则；<br>（6）GA/T 686－2007 信息安全技术虚拟专用网安全技术要求 |
| 内容安全 | 指对信息在网络内流动中的选择性阻断，以保证信息流动的可控能力。主要涉及信息的机密性、真实性、可控性、可用性等 | （1)采用文本识别、图像识别、流媒体识别、群发邮件识别等：用于对信息的理解与分析；<br>（2）采用面向内容的过滤技术（CVP）、面向 URL 的过滤技术（UFP）、面向 DNS 的过滤技术等：用于对信息的过滤；<br>（3）运用数据挖掘技术：发现信息；<br>（4）部署针对即时通、MSN 等应用协议的分析技术：对特定协议的理解；<br>（5)采用 VoIP 识别技术：对数字化语音信息的理解和分析；<br>（6）采用音频识别与按内容匹配：锁定音频目标进行 | 目前国家暂无内容安全相关的制度和标准，只有与音、视频，网络协议相关的网络标准，如 RFC 等 |

续表

| 安全层面 | 含　义 | 安全任务列表 | 安全管理依据的相关制度和法规 |
|---|---|---|---|
| 管理安全 | 在信息安全的保障过程中，除上述技术保障之外的与管理相关的人员、制度和原则方面的安全措施 | （1）设置独立的安全管理和审计人员：设置安全人员有助于安全工作的开展和全局安全掌控；<br>（2）权限分离：对不同的系统和应用设定与业务无关的安全人员参与，做到权限分离，最大程度地保证企业安全运维；<br>（3）安全制度：设立健全的企业安全管理和运维制度，保证企业安全运维；<br>（4）安全培训：通过培训提高企业人员的安全意识和安全技能，做到有备无患 | （1）BS7799；<br>（2）ISO/IEC17799；<br>（3）ISO/IEC27001 等 |

## 3.5 规划企业 Linux 安全的实施内容

操作系统是控制其他程序运行、管理系统资源并为用户提供操作界面的系统软件的集合。操作系统身负诸如管理与配置内存、决定系统资源供需的优先次序、控制输入与输出设备、操作网络与管理文件系统等基本事务。操作系统管理计算机系统的全部硬件资源包括软件资源及数据资源；控制程序运行；改善人机界面；为其他应用软件提供支持等，使计算机系统所有资源最大限度地发挥作用，为用户提供方便的、有效的、友善的服务界面。

作为企业运维应用最为广泛，软硬件均非常依赖的操作平台，操作系统在企业信息安全的保障中也有着非常重要的作用。该层次的安全问题来自网络内使用的操作系统的安全，当前的以Windows 系列为主导的操作系统从根本上来说都存在着相当大的安全性问题，非常容易遭受因为自身漏洞而导致的黑客攻击。这具体表现在两个大的方面，一是操作系统本身的缺陷带来的不安全因素，主要包括身份认证、访问控制、系统漏洞等；二是对操作系统的安全配置问题，主要包括密码问题、应用服务的配置问题等。就第一个问题来说，目前的企业大多采用优秀的网络操作系统 Linux 来替代 Windows，可以极大地减少系统漏洞带来的网络威胁，同时由于其是开源软件，学术界和企业界都以其为蓝本进行安全操作系统的开发，相信不久将会有较为成型的安全操作系统能够进入企业和广大用户的家庭；第二个问题是所有操作系统都存在的问题，具体需要通过提高使用人员的自身素质和安全防范意识来解决，设置安全的密码和合理配置网络应用服务，关掉不必要的“后门”，使得企业安全在操作系统这一层能够稳固。

因此，结合 3.4.3 小节介绍的企业信息安全的范围和实施手段，该 500 强企业将企业 Linux安全实施的工作界定为以下两个方面。

（1）运行安全。主要涉及操作系统自身系统安全及网络层安全。

- 操作系统安全。主要体现在如何通过用户管理、操作系统增强技术、密码管理、主机入侵检测系统技术等来保障操作系统自身安全，为其上的应用提供有力的安全保障和支撑。
- 网络层安全。该层次的安全问题主要体现在网络方面的安全性，而且体现在以 Linux 为平台基础的运行的网络服务上，包括网络层身份认证、网络资源的访问控制、DNS 域名系统的安全、路由系统的安全、防火墙技术、入侵检测技术、网络设施防病毒等。该层的安全及安全技术问题是当前企业面临的最大问题，由于互联网的开放性和普遍性，企业为了获得经济效益及提升知名度，不得不通过网络这个手段来进行许多商业宣传和运作活动，那么不可避免地会为网络威胁所困扰，当前的网络层面的 DoS 攻击、DDoS 攻击等危害日益增大，且追踪和防范的难度也越来越大，它们针对 TCP/IP 协议栈在设计和实现上的一些漏洞进行攻击，从而使得被攻击目标出现故障或者是“瘫痪”，该层的安全防护技术主要是针对各种类型的 DoS 和 DDoS 攻击进行防护和抑制。

（2）数据安全。指对在 Linux 系统中，信息在数据收集、处理、存储、检索、传输、交换、显示、扩散等过程中的保护，使得在数据处理层面保障信息依据授权使用，不被非法冒充、窃取、篡改、抵赖。主要涉及信息的机密性、真实性、完整性、不可抵赖性等。主要包括加解密技术、VPN 技术等。

# 第 4 章

# 选型：企业 Linux 软硬件选型及安装部署

## 本章导读

500 强企业在选择企业操作系统的时候非常地考究。一是要选择安全、可靠的系统平台来构建自己的应用；二是要考虑到能够培养一批熟练和高级的系统管理人员；三是要注意节省开支，尽量“花小钱，办大事”，而不是外界所误认为的不管三七二十一，选贵的，用好的，结果都不是自己需要的。

对于 Linux 操作系统来说，企业在选择时要考虑到其发行套件、内核版本以及服务器选型等诸多问题。并且，为了节省人力、物力，需要大规模地高效部署 Linux，如何解决这些问题呢？本章将为读者一一解答。

# 4.1 Linux 应用套件选择

## 4.1.1 Linux 的历史

要讲 Linux 的发展历史，不能不提到 UNIX 和 Minix。UNIX 的早期版本源代码可以免费获得，但是当 AT&T 发布版 UNIX 7 时，开始认识到 UNIX 的商业价值，于是发布的版本 7 许可证禁止在课程中研究其源代码以免其商业利益受到损害。许多学校为了遵守该规定，就在课程中略去 UNIX 的内容而只讲操作系统理论。

只讲理论使学生对实际的操作系统产生片面的认识。为了扭转这种局面，坦尼鲍姆决定编写一个在用户看来与 UNIX 完全兼容，然而内核全新的操作系统——Minix。坦尼鲍姆希望通过 Minix 读者可以剖析一个操作系统，研究其内部如何运作。Minix 其名称源于“Mini-UNIX”。Minix 一直恪守“Small is beautiful”的原则，其最早的版本甚至不需要硬盘就可以运行，这使得许多早期的学生都能够有能力负担其硬件的要求。随着其功能和规模的增长，大多数人都想在 Minix 中加入一些新特性以使之更大、更有用，而 Minix 的作者在几年内也一直坚持不增加新特性，使 Minix 保持短小精悍的特点，便于学生理解。此后，芬兰学生 Linus Torvalds 决定编写一个类似 Minix 的系统，其特征繁多且面向实用而非教学。他编写的这个操作系统就是 Linux。

1990 年，Linus Torvalds 用汇编语言写了一个在 80386 保护模式下处理多任务切换的程序，后来从 Minix 中得到灵感，他开始写了一些硬件的设备驱动程序、一个小的文件系统，这样 0.0.1 版本的 Linux 就出来了，但是它必须在有 Minix 的机器上编译以后才能运行。这时候，Linus 决定彻底抛弃 Minix，编写一个完全独立的操作系统。1991 年 10 月 5 号 Linus 发布了 Linux 0.0.2 版本，这个版本已经可以运行 bash（一种用户与操作系统内核通信的命令解释软件）和 GCC（GNU C 编译器）了。

Linus 从一开始就决定自由扩散 Linux。他把源代码发布在网上，随即就引起爱好者的注意，他们通过互联网也加入了 Linux 的内核开发工作，一大批高水平程序员的加入，使得 Linux 迅猛发展。到 1993 年底，Linux 1.0 终于诞生。Linux 1.0 是一个功能完备的操作系统，其内核写得紧凑高效，可以充分发挥硬件的性能，在 4MB 内存的 80386 机器上也表现得非常好。

Linux 从 1.3 版本之后开始向其他硬件平台上移植。目前 Linux 能将硬件的性能充分发挥出来，可以囊括从低端到高端的所有应用。现在 Linux 可以在 Intel、DEC 的 Alphas、Motorola 的 M68k、Sun Sparc、PowerPC、MIPS 等处理器上运行。

Linux 虽然加入 GNU 并遵循 GPL，但是并不排斥商家的参与，不排斥在 Linux 上开发商业软件，故而使 Linux 开始了新的飞跃，出现了很多的 Linux 发行版，如 Slackware、Red Hat、Suse、TurboLinux、OpenLinux 等 10 多种，而且仍在增加。许多公司还在 Linux 上开发商业软件或把其他 UNIX 平台的软件移植到 Linux 上来。如今很多 IT 界的大腕如 IBM、Intel、Oracle、Infomix、Sysbase、Corel、CA、Novell 等都宣布支持 Linux。商家的加盟弥补了纯自由软件的不足和发展障碍，使 Linux 得以迅速普及。

## 4.1.2 与 Linux 相关的基本概念

### 1. 开源软件

开放源代码软件是一种公开源代码的软件，任何人都可以修改、使用、拷贝、分发软件的源

代码。Linux 和 Zope 是最典型的开放源代码软件。

开放源代码软件由众多的商业公司共同开发，能够得到更好的质量保证。Zope 是由数百个公司、数千个开发人员共同维护的，他们共同组成了 Zope 社区 。同样 Plone 也是由很多的商业公司在共同开发和维护，润普公司正是这些商业公司的积极参与者之一。其实，IBM 和 HP 也都是支持开放源代码的软件公司。

同传统的封闭源代码软件相比，开放源代码软件为客户带来如下价值。

- 没有版权问题。客户无需支付软件版权费用，便可授权使用。这同时极大地降低了解决方案的成本。
- 更加安全和稳定。由于其开放性，开放软件源代码可以得到全世界众多同行的审查，因此更易具备类似 Linux 的安全性和稳定性
- 更强的生命力：开放源代码产品有更多的用户，因此它有着更强的生命力。开放源代码软件不会因某个具体的公司的倒闭而结束。

### 2. GNU

GNU 是 GNU’s Not Unix（网站为：www.gnu.org）的递归缩写，是由自由软件大师 Richard Stallman 在 1983 年 9 月 27 日公开发起的。它的目标是创建一套完全自由的操作系统。

为保证 GNU 软件可以自由地使用、复制、修改和发布，所有 GNU 软件都在一份禁止其他人添加任何限制的情况下授权所有权利给任何人的协议条款，GNU 通用公共许可证（GNU General Public License，GPL）。这个就是被称为“反版权”（或称 Copyleft）的概念。

1985 年 Richard Stallman 又创立了自由软件基金会（Free Software Foundation）来为 GNU 计划提供技术、法律以及财政支持。尽管 GNU 计划大部分时候是由个人自愿无偿贡献，但 FSF 有时还是会聘请程序员帮助编写。当 GNU 计划开始逐渐获得成功时，一些商业公司开始介入开发和技术支持。其中最著名的就是之后被 Red Hat 兼并的 Cygnus Solutions。

到了 1990 年，GNU 计划已经开发的软件包括了一个功能强大的文字编辑器 Emacs，C 语言编译器 GCC，以及大部分 UNIX 系统的程序库和工具。

1991 年 Linus Torvalds 编写出了与 UNIX 兼容的 Linux 操作系统内核并在 GPL 条款下发布。Linux 之后在网上广泛流传，许多程序员参与了开发与修改。1992 年 Linux 与其他 GNU 软件结合，完全自由的操作系统正式诞生。(尽管如此 GNU 计划自己的内核 Hurd 依然在开发中，目前已经发布 Beta 版本）。因此，严格的说，Linux 应该称为 GNU/Linux。并且，许多 UNIX 系统上也安装了 GNU 软件，因为 GNU 软件的质量比之前 UNIX 的软件还要好。GNU 工具还被广泛地移植到 Windows 和 Mac OS 上。



图 4-1　GNU LOGO

图 4-1 给出了 GNU 的常用图标。

### 3. GPL

GNU 通用公共许可证（GNU General Public License，简称为 GPL）是由自由软件基金会发行的用于计算机软件的许可证（网站为：http://www.gnu.org/licenses/gpl.html）。最初由 Richard Stallman 为 GNU 计划而撰写。目前大多数的 GNU 程序和超过半数的自由软件使用此许可证。此许可证最新版本为“版本 3”，2007 年 6 月发布。

GPL 不会授予许可证接受人无限的权利。再发行权的授予需要许可证接受人开放软件的源代码，及所有修改。且复制件、修改版本，都必须以 GPL 为许可证。这些要求就是 Copyleft，它的基础就是作品在法律上版权所有。由于它版权所有，许可证接受人就无权进行修改和再发行（除合理使用），除非它有一个 Copyleft 条款。如果某人想行使通常被法律所禁止的权利，只需同意 GPL 的条款。相反地，如果某人发行软件违反了 GPL（比如不开放源代码），他就有可能被原作者起诉。Copyleft 利用版权法来达到与其相反的目的：Copyleft 给人不可剥夺的权利，而不是版权法所规定的诸多限制。这也是 GPL 被称作“被黑的版权法”的原因。 许多 GPL 软件发行者都把源代码与可执行程序捆绑起来。另一方式就是以物理介质（比如 CD）为载体提供源代码。在实践中，许多 GPL 软件都是在互联网上发行的，源代码也有许多可以 FTP 方式得到。Copyleft 只在程序再发行时发生效力。对软件的修改可以不公开或开放源代码，只要不发行。注意 Copyleft 只对软件有效力，而对软件的输出并无效力（除非输出的是软件本身）。不过这在 GPL 版本 3 中可能会有改动。

GPL 设计为一种许可证，而不是合同。在英美法系国家，许可证与合同有法律上的明确区别：合同由合同法保障效力，而 GPL 作为一种许可证有版权法保障效力。不过在许多采用欧陆法系的国家并无此种区别。GPL 原理简单：在版权法下，你不遵守 GPL 的条款和条件就没有相应权利。而作品在没有 GPL 的情况下，版权法作为默认条款发生效力，而不是作品进入公有领域。

4. POSIX

POSIX 表示可移植操作系统接口（Portable Operating System Interface，缩写为 POSIX）。电气和电子工程师协会（Institute of Electrical and Electronics Engineers，IEEE）最初开发 POSIX 标准，是为了提高 UNIX 环境下应用程序的可移植性。然而，POSIX 并不局限于 UNIX。许多其他的操作系统，例如 DEC OpenVMS 和 Microsoft Windows NT 都支持 POSIX 标准，尤其是 IEEE Std. 1003.1-1990（1995 年修订）或 POSIX.1。POSIX.1 提供了源代码级别的 C 语言应用编程接口（API）给操作系统的服务程序，例如读写文件。POSIX.1 已经被国际标准化组织（International Standards Organization，ISO）所接受，被命名为 ISO/IEC 9945-1:1990 标准。

POSIX 现在已经发展成为一个非常庞大的标准族，某些部分正处在开发过程中。而 Linux 的目标是保持和 POSIX 的兼容。

### 4.1.3 Linux 的主要特点

Linux 从一个由个人开发的操作系统雏形经过短短十多年时间就发展成为今天举足轻重的操作系统，与 Windows、UNIX 一起形成操作系统领域三足鼎立的局势，必定有其原因。Linux 自身的特点就是其获得成功的原因。Linux 具有以下特性。

- 公开源码。作为程序员，通过阅读 Linux 内核和 Linux 下其他程序的源代码，可以学到很多编程经验和其他知识；作为最终用户，使用 Linux 避免了使用盗版 Windows 的尴尬，也避免了使用正版 Windows 的庞大费用。一个比较著名的例子是，墨西哥政府采用 Linux 操作系统替代使用 Windows 操作系统，大约节省了 1.24 亿美元。
- 系统稳定。Linux 采用了 UNIX 的设计体系，汲取了 UNIX 系统 25 年的发展经验。Linux 操作系统体现了现代操作系统的设计理念和最经得住时间考验的设计方案。在服务器操作系统市场上，Linux 已经超过 Windows 成为服务器首选操作系统。

- 性能突出。德国 C'T 最近公布了最新的 Windows 和 Linux 之间的测试结果。测试是由 Jurgen Schmidt 组织的，结果表明两种操作系统在各种应用情况下，尤其是在网络应用环境中，Linux 的总体性能更好。
- 设备独立性。设备独立性是指操作系统把所有外部设备统一当作文件来看待，只要安装它们的驱动程序，任何用户都可以像使用文件一样，操纵、使用这些设备，而不必知道它们的具体存在形式。Linux 是具有设备独立性的操作系统，它的内核具有高度适应能力，随着更多的程序员加入 Linux 编程，会有更多硬件设备加入到各种 Linux 内核和发行版本中。另外，由于用户可以免费得到 Linux 的内核源代码，因此，用户可以修改内核源代码，以便适应新增加的外部设备。
- 安全性强。各种病毒的频繁出现使得微软几乎每隔几天就要为 Windows 公布补丁。而现在针对 Linux 的病毒则非常少，而且 Linux 的公开源代码的开发方式使得各种漏洞都能够在 Linux 上得到及早发现和弥补。
- 跨平台。Windows 只能在 Intel 构架下运行，但是 Linux 除了可以运行于 Intel 平台外，还可以运行于 Motorola 公司的 68K 系列 CPU，IBM、Apple、Motorola 公司的 PowerPC CPU，Compaq 和 Digital 公司的 Alpha CPU、MIPS 芯片，Sun 公司的 SPARC 和 UltraSparc CPU，Intel 公司的 StrongARM CPU 等处理器系统。
- 完全兼容 UNIX。Linux 和现今的 UNIX、System V、BSD 等三大主流的 UNIX 系统几乎完全兼容，在 UNIX 下可以运行的程序，完全可以移植到 Linux 下运行。
- 良好的可移植性。Linux 是一种可移植的操作系统，能够在从微型计算机到大型计算机的任何环境中和任何平台上运行。可移植性为运行 Linux 的不同计算机平台与其他任何机器进行准确而有效的通信提供了手段，不需要另外增加特殊的和昂贵的通信接口。
- 强大的网络服务。Linux 诞生于因特网，它具有 UNIX 的特性，保证了它支持所有标准因特网协议，而且 Linux 内置了 TCP/IP 协议。事实上，Linux 是第一个支持 IPv6 的操作系统。

### 4.1.4 Linux 的应用领域

Linux 从诞生到现在，已经在各个领域得到了广泛应用，显示了强大的生命力，并且其应用正日益扩大。下面列举其主要应用领域。

- 教育领域。设计先进和公开源代码这两大特性使 Linux 成为操作系统课程的好教材。
- 网络服务器领域：稳定、健壮、系统要求低、网络功能强使 Linux 成为现在 Internet 服务器操作系统的首选，现已达到了服务器操作系统市场 25%的占有率。
- 企业 Intranet。利用 Linux 系统可以使企业用低廉的投入架设 E-mail 服务器、WWW 服务器、代理服务器、透明网关、路由器。
- 视频制作领域。著名的影片《泰坦尼克号》就是由 200 多台装有 Linux 系统的机器协作完成其特技效果的。

### 4.1.5 常见的 Linux 发行套件

1. Red Hat Linux/ Red Hat Enterprise Linux

Red Hat Linux，俗称红帽子 Linux，是应用最广、最为成熟的 Linux 发行版本，同时也可以

说是最著名的 Linux 版本了，Red Hat Linux 已经创造了自己的品牌，越来越多的人听说过它（请参见图 4-2 所示的 Red Hat Linux 的经典图标）。Red Hat 在 1994 年创业，当时聘用了全世界 500 多名员工，他们都致力于开放的源代码体系。



图 4-2　Red Hat Linux 图标

Red Hat Linux 是一个高效的商用版本，是喜爱 Linux 的高手们进行应用级和内核级开发的最合适的选择，同时也是初学者步入 Linux 这个自由而神圣的殿堂的最佳使用工具。它享有极好的口碑，同时它的个人版本和企业版本在 Linux 个人用户和服务器市场都占有极高的比率。同时，其丰富的系统管理和网络管理工具，使得用户越来越轻松方便地对系统和网络平台进行维护和管理。同时，由于其用户是一个巨大的群体，因此作为开源操作系统，很多的应用软件也针对其被开发和共享出来，因此也为该系统的发展提供了很多便利。

Red Hat Enterprise Linux（缩写为 RHEL，Red Hat 的企业版）。Red Hat 现在将企业发展的重心转向了服务器版的 Linux 开发，在版本上注重了性能、稳定性以及对硬件的支持。由于企业版操作系统的开发周期较长，注重性能、稳定性和服务端软件支持，因此版本更新相对较缓慢。一般分为三个版本 AS（Advanced Server）、ES（Entry Server）和 WS（Workstation Server）。2010 年 11 月 10 日，Red hat 公司正式发行了 RHEL 6 版本。

### 2. Fedora Core/Fedora

Fedora Core/Fedora 是众多 Linux 发行套件之一（请参见图 4-3 所示的经典图标）。它是一套从 Red Hat Linux 发展出来的免费 Linux 系统。Fedora Core 的前身就是 Red Hat Linux。2003 年 9 月，红帽公司（Red Hat）突然宣布不再推出个人使用的发行套件而专心发展商业版本（Red Hat Enterprise Linux）的桌面套件，但是红帽公司也同时宣布将原有的 Red Hat Linux 开发计划和 Fedora 计划整合成一个新的 Fedora Project。Fedora Project 将会由红帽公司赞助，以 Red Hat Linux 9 为范本加以改进，原本的开发团队将会继续参与 Fedora 的开发计划，同时也鼓励开放原始码社群参与开发工作。自 Fedora Core 6 之后的版本都更名为 Fedora（如 Fedora 7）。现在 Fedora 最稳定的版本是 Fedora 10（Fedora 11 仍旧处于 Beta 测试版本阶段）。

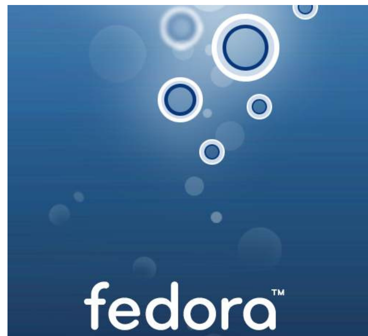


图 4-3　Fedora Core/Fedora Linux 图标

Fedora Core/Fedora 无论从稳定性、口碑以及可管理性方面，都极大地继承了以前 Red Hat Linux 的优良传统，所谓“青出于蓝而胜于蓝”，目前它已经成为 Linux 操作系统众多发行套件中最炙手可热的一种。

### 3. CentOS

CentOS（Community Enterprise Operating System）是一个基于 Red Hat 企业级 Linux 提供的可自由使用的源代码企业级的 Linux 发行版本。每个版本的 CentOS 都会获得七年的支持（通过安全更新方式）。新版本的 CentOS 每两年发行一次，而每个版本的 CentOS 会定期（大概

每六个月）更新一次，以便支持新的硬件。这样，建立了一个安全、低维护、稳定、高预测性、高重复性的 Linux 环境。

CentOS 是 Linux 发行版之一，它是来自于 Red Hat Enterprise Linux 依照开放源代码规定释出的源代码所编译而成。由于出自同样的源代码，因此有些要求高度稳定性的服务器以 CentOS 替代商业版的 Red Hat Enterprise Linux 使用。两者的不同在于 CentOS 并不包含封闭源代码软件。

CentOS 是一个开源软件贡献者和用户的社区。它对 RHEL 源代码进行重新编译，成为众多发布新发行版本的社区当中的一个，并且在不断的发展过程中，CentOS 社区不断与其他的同类社区合并，使 CentOS Linux 逐渐成为使用最广泛的 RHEL 兼容版本。CentOS Linux 的稳定性不比 RHEL 差，唯一不足的就是缺乏技术支持，因为它是由社区发布的免费版。

CentOS Linux 与 RHEL 产品有着严格的版本对应关系，例如使用 RHEL4 源代码重新编译发布的是 CentOS Linux 4.0，RHEL5 对应的是 CentOS Linux 5.0，RHEL6 对应的是 CentOS Linux 6。由于 RHEL 产品的生命周期较长（通常具有 3～5 年的官方支持），因此 Red Hat 公司在 RHEL 系列产品发布后每隔一段时间，都会将累积的更新程序重新打包成为更新的发行版进行发布，通常称为 RHEL Update。例如，RHEL5 的第一个更新版本叫做 RHEL 5 Update 1，用户通常也称为 RHEL 5.1。对 Red Hat 公司发布的每一个 RHEL Update CentOS 社区都会发布对应的更新发行版，例如根据 RHEL 5 的 Update 1 更新程序源码包，CentOS 会重新编译并打包发布 CentOSLinux 5.1 版。CentOS Linux 和与之对应版本号的 RHEL 发行版具有软件包级别的二进制兼容性，即某个 RPM 软件包如果可以安装运行在 RHEL 产品中，就可以正常地安装运行在对应版本的 CentOS Linux 中。CentOS Linux 由于同时具有与 RHEL 的兼容性和企业级应用的稳定性，又允许用户自由使用，因此得到了越来越广泛的应用。

CentOS 的主要特点如下。

- 可以把 CentOS 理解为 Red Hat AS 系列！它完全就是对 Red Hat AS 进行改进后发布的！各种操作、使用和 Red Hat 没有区别！
- CentOS 完全免费，不存在 Red Hat AS4 需要序列号的问题。
- CentOS 独有的 yum 命令支持在线升级，可以即时更新系统，不像 Red Hat 那样需要花钱购买支持服务！
- CentOS 修正了许多 Red Hat AS 的 BUG！
- CentOS 版本说明： CentOS3.1 等同于 Red Hat AS3Update1，CentOS3.4 等同于 Red Hat AS3 Update4，CentOS4.0 等同于 Red Hat AS4。

虽然说 CentOS 只是 RHEL 的克隆、CentOS 社区发生过险些解散的危机，但是我们不得不承认 CentOS 确实是一个相当不错的 Linux 企业级发行版，至少它满足了企业的需求，还为企业节省了一笔资金。不过作为企业的技术负责人来讲，在选择操作系统的时候还需要经过深思熟虑，毕竟企业需要长久发展。



图 4-4 CentOS 图标

如图 4-4 所示是 CentOS 的图标。

4. Debian

Debian 拥有很长的发展历史，Debian Project 诞生于 1993 年 8 月 13 日，它的目标是提供一

个稳定容错的 Linux 版本。截至目前，Debian 最新的正式发行版版本号是 4.0，其最后一次更新发生在 2008 年 10 月 23 日。在发展初期，支持 Debian 的不是某家公司，而是许多在其改进过程中投入了大量时间的开发人员，这种改进吸取了早期 Linux 的经验，因而发展也较为迅猛。图 4-5 所示 Debian 的图标。



图 4-5　Debian Linux 图标

Debian 以其稳定性著称，虽然它的早期版本 Slink 有一些问题，但是它的现有版本 Potato 已经相当稳定了。这个版本更多地使用 pluggable authentication modules（PAM），综合了一些更易于处理的需要认证的软件（如 winbind for Samba）。在众多 Linux 发行版中，Debian Linux 的显著特点是拥有十分完善易用的包管理机制，非常容易维护。另外，Debian Linux 由开源社区维护，免费发行，是一个开放源代码的操作系统。它由许多志愿者维护，是真正的非商业化 Linux。

另外，Debian Linux 是最流行的 Linux 发行版之一。基于 Debian，衍生了众多其他的分支 Linux 发行版本。Debian GNU/Linux 不单是个操作系统，人们真正需要的是应用软件，也就是帮助他们完成工作的程序，因此，从文档编辑，到电子商务，到游戏娱乐，到软件开发，Debian 为用户带来了超过 18733 个软件包（为了能在用户的机器上轻松地安装，这些软件包都已经被编译包装为一种方便的格式），并且这些全部都是自由软件。

Debian 主要通过基于 Web 的论坛和邮件列表来提供技术支持。作为服务器平台，Debian 提供一个稳定的环境。为了保证它的稳定性，开发者不会在其中随意添加新技术，而是通过多次测试之后才选定合适的技术加入。

### 5. Ubuntu

Ubuntu 由马克・舍特尔沃斯创立，其首个版本于 2004 年 10 月 20 日发布，并以 Debian 为开发蓝本。目前其最新版本为 8.0.4（请参见图 4-6 所示的 Ubuntu 图标）。其以每六个月发布一次新版本为目标，因此使得人们能够得以更快捷和迅速地获取新软件和享受心得功能特性。Ubuntu 的每个新版本均会包含最新版本的 GNOME 桌面环境，并且会在 GNOME 发布新版本后一个月内发行。与以往基于 Debian 的 Linux 发行版（如 MEPIS、Xandros、Linspire、Progeny 与 Libranet）等比较起来，Ubuntu 更接近 Debian 的开发理念，因为其主要使用自由与开源软件。


图 4-6　Ubuntu Linux 图标

Ubuntu 的软件套件主要是基于 Debian 的不稳定分支进行开发。在开发过程中，Ubuntu 会将所有对软件套件的修改及时向 Debian 作出回馈。并且，很多 Ubuntu 的开发者均为 Debian 的主要软件套件的维护者。然而，Debian 与 Ubuntu 的软件套件并不一定与对方兼容。

Ubuntu 十分注重系统的安全性，它采用 Sudo 工具，规定所有系统相关的任务均需使用此指令并输入密码才能进行下一步工作，因此比起传统以登录系统管理员账号进行管理工作具有更好的安全性。同时，Ubuntu 也非常注重系统的可用性，它的设计目标是在 Ubuntu 系统标准安装完

成后就可以为使用者立刻投入使用。因此，从开发流程和设计思路上来看，Ubuntu 都是 Debian 的改进版本，且相对有较高的稳定性、可用性和管理性。

### 6. SuSE Linux

SuSE Linux 原是以 Slackware Linux 为基础，并提供完整的使用界面的产品。1992 年 Peter McDonald 发布了 Softlanding Linux System（SLS）发行版。这套发行版包含的软件非常多，更首次收录了 X Window 及 TCP/IP 等套件。Slackware 就是一个基于 SLS 的发行版。SuSE 于 1992 年末创办，目的是成为 UNIX 技术公司，专门是为德国人推出量身定做的SLS/Slackware软件及UNIX/Linux说明文件。1994 年，他们首次推出了 SLS/Slackware 的安装光碟，命名为 S.u.S.E. Linux 1.0。其后，它又进一步综合了 Florian La Roche 的 Jurix distribution（也是一个基于 Slackware 的发行版），于 1996 年推出一个完全自家打造的发行版本——S.u.S.E. Linux 4.2。其后 SuSE Linux 汲取和采纳了 Red Hat Linux 的不少特点和技术。请参见图 4-7 所示的 SuSE Linux 图标。

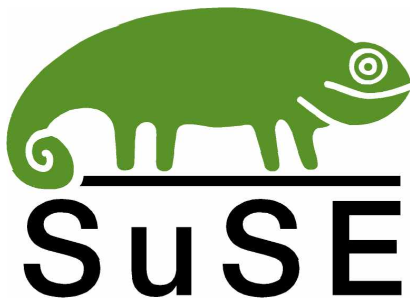


图 4-7　SuSE Linux 图标

Novell 日前发布了 SuSE Linux 的最新版本 10.1，这是 SuSE Linux 与开源社区完全合作以来产生的第一个版本，提供了一切为用户开始掌握 Linux 所准备的功能。10.1 中加入了所有最新的 Linux 功能，包括桌面搜索、图形界面和 Windows 程序兼容性等许多技术。无论是用于个人网络服务器、开发环境，还是基本的计算功能，用户都能在其中发现自己所需要的技术和应用。10.1 中包含了 1500 多个软件包，为用户提供完整的桌面效果，包括 OpenOffice.org 套件、Firefox 浏览器、GAIM 即时通讯客户端、多媒体工具、图像处理软件、最新的虚拟机软件，以及簇状文件系统技术等。

### 7. Mandriva

Mandriva Linux 的前身是欧洲最大的 Linux 厂商之一 Mandrakesoft 的产品 Mandrake LinuxTemplate:Fact。Mandrake Linux 早期方便的字体安装工具和默认的中文支持，为 Linux 普及作出了很大的贡献。但是 2004 年前后 Mandrakesoft 陷入财务危机，濒临破产。公司于 2005 年 2 月 24 日与拉丁美洲最大的 Linux 厂商 Conectiva 达成了收购协议，金额为 170 万欧元，约合 223 万美元，以股票形式交易，新公司 Mandriva 旗下品牌 Mandrake Linux 更名为 Mandriva Linux。其实最早 Mandrake 的开发者是基于 Red Hat 进行开发的。Red Hat 默认采用 GNOME 桌面系统，而 Mandrake 将之改为 KDE。而由于当时的 Linux 普遍比较难安装，不适合第一次接触 Linux 的新手，所以 Mandrake 还简化了安装系统。

Mandriva 的官方网站是 http://www.mandriva.com/，其标志性图标如图 4-8 所示。



图 4-8　Mandriva 图标

## 4.1.6　企业的选择：Fedora vs Red Hat Enterprise Linux

Red Hat Linux 系列版本相互补充，相得益彰。如果第一次接触 Linux，那么建议首先使用 Fedora Core，Fedora Core 安装简单，对硬件支持很好，界面华丽，适合习惯 Windows 操作习惯

的用户接受，同时也可以体验 Red Hat Linux 的最新功能；如果对 Linux 有一定的了解，需要深入学习，建议使用 CentOS 系统，因为可以随意在网上获取安装程序，不需要授权和注册，功能和 RHEL 基本完全类似；如果是在做 Linux 企业级应用和部署，建议使用 RHEL 版本。

## 4.2 Linux 内核版本选择

严格地说，Linux 是在 GPL（GNU General Public License）版权协议下发行的操作系统内核，其版权属于 Linus Torvalds。我们通常所说的 Linux 是指包含 kernel（内核）、utilities（系统工具程序）以及 application（应用软件）的一个完整的操作系统，它实际上是 Linux 的发行版本，是某些公司或组织把 Linux 内核、源代码以及相关的应用程序组织在一起发行。国际上比较著名的 Linux 发行版本有 Red Hat、SlackWare、Debian、Fedora、Ubuntu 等。国内也有不少 Linux 的发行版本，其中最为著名的首推北京中科红旗软件技术有限公司发布的红旗 Linux。

Linux 是 UNIX 的“克隆”，在源代码级上兼容绝大部分的 UNIX 标准（如 IEEE POSIX、System V、BSD 等），并且符合 POSIX 规范。例如对于 System V 来说，把其上的程序源代码拿到 Linux 下重新编译后就可以运行，而对于 BSD UNIX 来说，它的可执行文件可以直接在 Linux 环境下运行。

需要说明的是，GPL 同其他的自由软件许可证一样，许可公众享有运行、复制软件的自由，发行传播软件的自由，获得软件源码的自由，改进软件并将自己做出的改进版向社会发行传播的自由。

由于 Linux 的源程序是完全公开的，任何人只要遵循 GPL，就可以对内核加以修改并发布给他人使用。Linux 内核的版本在发行上有自己的规则，可以从其版本号加以识别。版本号的格式为“x.yy.zz”。其中 x 介于 0 ～9 之间，而 yy，zz 则介于 0～99 之间。通常数字越大说明版本越高。而且它有一个非常简单的编号约定：任何偶数的核心（例如 2.0.30）都是一个稳定地发行的核心，而任何奇数的核心（例如 2.1.42）都是一个开发中的核心，用以进行最新功能的测试，不建议初学者和生产过程中使用。一些版本号后面有时会见到 pNN 的字样，NN 是介于 0～20 之间的数字，它代表对某一版本的内核“打补丁”或者是修改的次数。

Linux 内核版本发布的官方网站是 http://www.kernel.org。新版本的内核分两种，一种是 full Source 版本，另外一种是 patch 文件，即补丁。完整的内核版本比较大，一般是 tar.gz 或者是.bz2 文件，二者分别是使用 gzip 或者 bzip2 进行压缩的文件，使用时需要解压缩。patch 文件则比较小，一般只有几十 KB 到几百 KB，但是 patch 文件是针对于特定的版本的，用户需要找到自己对应的版本才能使用。

## 4.3 Linux 服务器选型

可以在 32 位的 Intel 和兼容的处理器架构（包括 AMD 处理器）以及 64 位处理器架构，例如 AMD64 处理器和使用英特尔 EM64T 技术的英特尔处理器上安装 Fedora/RHEL。在这些处理器架构上运行 Fedora/RHEL 可用的大部分硬件，你可以查看 hardware.redhat.com 的 Fedora/RHEL 的兼容和支持的硬件列表。虽然这些也涉及 RHEL，但它们对 Fedora 在什么上面运行将给予一个

很好的指导。发行说明还提供硬件信息。许多互联网网站讨论 Linux 硬件；使用 Google（www.google.com）搜索 Linux 的硬件支持、Fedora 的硬件或者你想了解更多 Linux 和特定的硬件信息（例如，Linux SATA 或 Linux a8n）。此外，许多 HOWTO 文件也涵盖了特定的硬件。Linux 硬件兼容性指南也可以参考，虽然它可能不是当时最新的。Fedora/RHEL 一般和 Windows 运行在同一系统，除非该系统包括一个很新的或不同寻常的组件。

运行 Fedora/RHEL 所需的硬件取决于你要设置成哪一种系统。一个运行文本（命令行）界面并装有极少软件包的很小系统，和一个运行 GUI 的系统、Apache Web 服务器系统、安装许多软件包的系统所需的硬件是有很大不同的。

网络连接对保持 Fedora/RHEL 版本的最新是非常重要的。声卡（或主板上的音效芯片）是不错的多媒体应用。如果你是在旧的或最小的硬件上安装 Fedora，并且要运行 GUI，可以考虑安装 LXDE（spins.fedoraproject.org/LXDE），因为它提供了一个轻量级的桌面，它比 GNOME 能够更有效地使用系统资源。

### 4.3.1　CPU（处理器）

Fedora 至少需要英特尔奔腾 Pro 处理器，它是对 i686 和以后的处理器进行了优化。建议文本系统的处理器速度最低为 200MHz、图形系统最低为 400MHz。RHEL 系统的信息请参阅 www.redhat.com/rhel/compare。

### 4.3.2　RAM（内存）

安装 32 位和 64 位的 Fedora 系统内存的要求是相同的。文本（命令行）系统至少需要 256MB 的 RAM，图形系统（台式机）的 RAM 需要 384～512MB。在某些情况下，安装一个很小的文本系统你可以用较少的 RAM。RHEL 系统的信息请参阅 www.redhat.com/rhel/compare。

运行一个 live CD 需要至少 256MB 的内存。

Linux 能够充分使用额外的内存：一个系统拥有更多的内存，它运行的速度就越快。扩展内存是性价比最好的 Linux 系统提速方法之一。

### 4.3.3　处理器架构

Fedora/RHEL CD 和 DVD 采用在一个特定的处理器架构（类处理器或 CPU）上进行编译的方案。下面列表描述了每个架构的 Fedora/RHEL 编译。每个架构处理器的详细清单参见 docs.fedoraproject.org/en-US/Fedora/15/html/Installation_Guide/ch-new-users.html#sn-which-arch。由于大家可获得 Linux 的源代码，一个具有一定水平的用户可以在其他处理器架构上编译运行 Fedora/RHEL。

如何确定在一个 64 位的处理器上应该安装 32 位的还是 64 位的 Fedora/RHEL？以下信息可能会帮助你做出决定，在一个 64 位的处理器上是要安装 32 位的还是 64 位的 Fedora/RHEL。

- EM64T/AMD64 处理器可以出色地运行任何版本的 Fedora/RHEL。
- 64 位分布式，可以使每个进程寻址超过 4 GB 的 RAM。更大的地址空间是 64 位分布式的最大优势。仅在针对某个工程/科学计算工作，或者当你运行多个虚拟机时，64 位分布式通常是有用的。

- 一般来说，一个 64 位处理器的速度并不比 32 位处理器的速度快，大多数基准显示或多或少性能相似。在某些情况下性能好，而在某些情况下性能糟：两种类型的处理器并没有明显的性能差异。
- 64 位 Linux 的内存模型，地址指针比 32 位 Linux 的大两倍。这种大小差异使得内存使用率增加超过 5%，这取决于运行的应用程序。如果系统内存不足，这种开销可能使性能恶化。
- ASLR（地址空间布局随机化）能够与 64 位 Fedora/RHEL 提供的更大地址空间更好地运行。ALSR 还可以帮助提高系统的安全性。参见 en.wikipedia.org/wiki/Address_space_layout_randomizationen.wikipedia.org/wiki/Address_space_layout_randomization。
- 64 位的 Fedora/RHEL，一些多媒体编码器的运行速度可提高 10%～30%。
- 由于更多的人正在使用 32 位 Linux，32 位 Linux 的错误比 64 位 Linux 的往往更易于发现和快速修复。
- 在 64 位的 Fedora/RHEL 上你可以和在 32 位系统一样只需点击一下，就可以设置 Flash Player 和 Java。然而对于某些应用，如 Skype，你必须应用困难的工作环境来在 64 位系统上运行它们。
- 专有的第三方应用程序的某些功能在 64 位架构上不可用。
- 没有简单的方法，在无须重新安装 Fedora/RHEL 的情况下，Fedora/RHEL 的 32 位和 64 位版本之间来回切换。
- 如果你不能确定使用哪种版本，请安装 32 位版本的 Fedora/RHEL。

i386/i686（Intel x86）

Fedora/RHEL32 位个人电脑 CD/DVD 上的软件是为 Intel x86 兼容处理器编译运行的，包括大部分 Intel 和 AMD 处理器，其中几乎所有的机器都在运行 Microsoft Windows，同时使用 Intel 处理器的还有一些较新的苹果 Macintosh 计算机。 N 和 Z 系列 Atom 处理器也是基于这种架构。如果你不知道机器的处理器是哪种类型的，那就假设它是这种类型的处理器。

x86_64（AMD64 和 Intel EM64T）

Fedora/RHEL 64 位 CD/DVD 上的软件是为 Athlon64 处理器编译运行的，包括 Athlon64、Opteron 和 Intel64 位处理器，它们采用 EM64T 技术，如 EMT64 至强。230 和 330 系列的 Atom 处理器也是基于这种架构的。

苹果的 PowerPC（PPC）

Fedora/RHEL 的 PPC 发行版运行于苹果 Macintosh G3、G4、G5、PowerBook 和基于 PPC 的 Macintosh 版本。

### 4.3.4 服务器类型选型

目前市面上的服务器琳琅满目，且分类方法也不一而足。企业用户在为自己的应用选择服务器时总是不知所措，无法做出合理的决策。下面将从架构和应用层次两方面来介绍当前服务器的主流分类方法，并介绍其特点，以帮助读者朋友来为自己的应用正确高效地选择服务器产品。

#### 1. 按照架构选择服务器

各种媒体上经常按塔式、机架式和刀片式这三种结构来划分服务器。服务器的外形为什么会

有这样的划分呢？主要原因就是具体的应用环境不同。

塔式服务器形似我们平时用的台式机，占用空间比较大，一般是一些小型企业自己使用、自己维护；

机架式服务器长得就像卧着的台式机，可以一台一台地放到固定机架上，因此而得名，它可以在专业的服务器托管提供商那里进行托管，这样每年只需支付一定的托管费，就免去了自己管理服务器的诸多不便。

刀片式服务器是近几年才比较流行的一种服务器架构，它非常薄，可以一片一片地叠放在机柜上，通过群集技术进行协同运算，能够处理大量的任务，特别适合分布式服务，如作为 Web 服务器。

1）塔式服务器

塔式服务器应该是大家见得最多，也最容易理解的一种服务器结构类型，因为它的外形以及结构都跟我们平时使用的立式 PC 差不多，当然，由于服务器的主板扩展性较强、插槽也多出一堆，所以个头比普通主板大一些，因此塔式服务器的主机机箱也比标准的 ATX 机箱要大，一般都会预留足够的内部空间以便日后进行硬盘和电源的冗余扩展。

由于塔式服务器的机箱比较大，服务器的配置也可以很高，冗余扩展更可以很齐备，所以它的应用范围非常广，应该说目前使用率最高的一种服务器就是塔式服务器。我们平时常说的通用服务器一般都是塔式服务器，它可以集多种常见的服务应用于一身，不管是速度应用还是存储应用都可以使用塔式服务器来解决。

就使用对象或者使用级别来说，目前常见的入门级和工作组级服务器基本上都采用这一服务器结构类型，一些部门级应用也会采用，不过由于只有一台主机，即使进行升级扩展也有个限度，所以在一些应用需求较高的企业中，单机服务器就无法满足要求了，需要多机协同工作，而塔式服务器个头太大，独立性太强，协同工作在空间占用和系统管理上都不方便，这也是塔式服务器的局限性。

2）机架式服务器

作为为互联网设计的服务器模式，机架式服务器是一种外观按照统一标准设计的服务器，配合机柜统一使用。可以说机架式是一种优化结构的塔式服务器，它的设计宗旨主要是为了尽可能减少服务器空间的占用，而减少空间的直接好处就是在机房托管的时候价格会便宜很多。

之所以说机架式服务器是作为为互联网设计的服务器模式，主要是因为很多专业互联网设备都是采用机架式的结构（多为扁平式），如交换机、路由器、硬件防火墙这些。这些设备之所以有这样一种结构类型，是因为它们都按国际机柜标准进行设计，这样大家的平面尺寸就基本统一，可一起安装在一个大型的立式标准机柜中。这样做的好处非常明显：一方面可以使设备占用最小的空间，另一方面则便于与其他网络设备的连接和管理，同时机房内也会显得整洁、美观。机架式服务器的宽度为 19 英寸，高度以 U 为单位（1U=1.75 英寸=44.45 毫米），通常有 1U、2U、3U、4U、5U、7U 几种标准的服务器。机柜的尺寸也是采用通用的工业标准，通常从 22U～42U 不等；机柜内按 U 的高度有可拆卸的滑动托架，用户可以根据自己服务器的标高灵活调节高度，以存放服务器、集线器、磁盘阵列柜等网络设备。服务器摆放好后，它的所有 I/O 线全部从机柜的后方引出（机架服务器的所有接口也在后方），统一安置在机柜的线槽中，一般贴有标号，便于管理。

现在很多互联网的网站服务器其实都是由专业机构（IDC）统一托管的，网站的经营者其实

只是维护网站页面，硬件和网络连接则交给托管机构负责，因此，托管机构会根据受管服务器的高度来收取费用，1U 的服务器在托管时收取的费用比 2U 的要便宜很多，这就是为什么这种结构的服务器现在会广泛应用于互联网事业。

相比较而言，机架式服务器因为空间比塔式服务器大大缩小，所以这类服务器在扩展性和散热问题上受到一定的限制，配件也要经过一定的筛选，一般都无法实现太完整的设备扩张，所以单机性能就比较有限，应用范围也比较有限，只能专注于某一方面的应用，如远程存储和 Web 服务的提供等，但由于很多配件不能采用塔式服务器的那种普通型号，而自身又有空间小的优势，所以机架式服务器一般会比同等配置的塔式服务器贵 20%～30%。

至于空间小而带来的扩展性问题，也不是完全没有办法解决。由于采用机柜安装的方式，因此多添加一个主机在机柜上是件很容易的事，然后再通过服务器群集技术就可以实现处理能力的增强。如果是采用外接扩展柜的方式也能实现大规模扩展，不过由于机架式服务器单机的性能有限，所以扩展之后也是单方面的能力得到增倍，所以这类服务器只是在某一种应用中比较出色，大家就把它划为功能服务器，这种服务器针对性较强，一般无法移做它用。

在价格方面，机架式服务器一般比同等配置的塔式服务器贵上二到三成。

3）刀片式服务器

对于企业和网络信息提供商来说，无限增长的数据必须集中存储和处理，于是未来的网络发展呈现出集中计算的趋势。集中管理模式与现有的分散管理模式，对服务器提出了新的要求：节约空间、便于集中管理、易于扩展和提供不间断的服务，成为对下一代服务器的新要求。

作为网络重要组成部分的服务器来说，性能已不仅仅是评价服务器的唯一指标了，用户更关心的是符合自己实际需要的产品。刀片式服务器是一种 HAHD（High Availability High Density，高可用高密度）的低成本服务器平台，是专门为特殊应用行业和高密度计算机环境设计的。其中每一块"刀片"实际上就是一块系统主板。

它们可以通过本地硬盘启动自己的如 Windows 2000/2003、Linux、Solaris 等，类似于一个个独立的服务器。在这种模式下，每一个主板运行自己的系统，服务于指定的不同用户群，相互之间没有关联。不过可以用系统软件将这些主板集合成一个服务器集群。

刀片式服务器比机架式服务器更节省空间，同时，散热问题也更突出，往往要在机箱内装上大型强力风扇来散热。此型服务器虽然较节省空间，但是其机柜与刀片价格都不低，一般应用于大型的数据中心或者需要大规模计算的领域，如银行电信金融行业以及互联网数据中心等。

目前，节约空间、便于集中管理、易于扩展和提供不间断的服务，成为对下一代服务器的新要求，而刀片式服务器正好能满足这一需求，因而刀片式服务器市场需求正不断扩大，具有良好的市场前景。

### 2．按照应用层次选择服务器

按应用层次划分通常也称为"按服务器档次划分"或"按网络规模"划分，是服务器最为普遍的一种划分方法，它主要根据服务器在网络中应用的层次（或服务器的档次）来划分的。要注意的是，这里所指的服务器档次并不是按服务器 CPU 主频高低来划分，而是依据整个服务器的综合性能，特别是所采用的一些服务器专用技术来衡量的。按这种划分方法，服务器可分为：入门级服务器、工作组级服务器、部门级服务器和企业级服务器。

1）入门级服务器

入门级服务器通常只使用一块CPU，并根据需要配置相应的内存（如256MB）和大容量IDE硬盘，必要时也会采用IDE RAID（一种磁盘阵列技术，主要目的是保证数据的可靠性和可恢复性）进行数据保护。入门级服务器主要是针对基于Windows NT、NetWare等网络操作系统的用户，可以满足办公室型的中小型网络用户的文件共享、打印服务、数据处理、Internet接入及简单数据库应用的需求，也可以在小范围内完成诸如E-mail、Proxy、DNS等服务。

对于一个小部门的办公需要而言，服务器的主要作用是完成文件和打印服务。文件和打印服务是服务器的最基本应用之一，对硬件的要求较低，一般采用单颗或双颗CPU的入门级服务器即可。为了给打印机提供足够的打印缓冲区需要较大的内存，为了应付频繁和大量的文件存取要求有快速的硬盘子系统，而好的管理性能则可以提高服务器的使用效率。

入门级服务器所连接的终端比较有限（通常为20台左右），况且在稳定性、可扩展性以及容错冗余性能较差，仅适用于没有大型数据库数据交换、日常工作网络流量不大、无须长期不间断开机的小型企业。不过要说明的一点就是，目前有的比较大型的服务器开发、生产厂商在后面我们要讲的企业级服务器中也划分出几个档次，其中最低档的一个企业级服务器档次就是称之为“入门级企业级服务器”，这里所讲的入门级并不是与我们上面所讲的“入门级”具有相同的含义，不过这种划分的还是比较少。还有一点就是，这种服务器一般采用Intel的专用服务器CPU芯片，是基于Intel架构（俗称“IA结构”）的，当然这并不是一种硬性的标准规定，而是由于服务器的应用层次需要和价位的限制。

2）工作组级服务器

工作组级服务器是一个比入门级高一个层次的服务器，但仍属于低档服务器之类。从这个名字也可以看出，它只能连接一个工作组（50台左右）那么多用户，网络规模较小，服务器的稳定性也不像后面我们要讲的企业级服务器那样高的应用环境，当然在其他性能方面的要求也相应要低一些，是一种支持特定的一组用户并为用户提供字处理和文件共享等功能的服务器，而这些服务只是某一人群所需要的。工作组级服务器具有以下几方面的特点。

- 通常仅支持单或双CPU结构的应用服务器（但也不是绝对的，特别是SUN就有能支持多达4个处理器的工作组级服务器，当然该类型的服务器价格方面也就有些不同了）。
- 可支持大容量的ECC内存和增强服务器管理功能的SM总线。
- 功能较全面、可管理性强，且易于维护。
- 采用Intel服务器CPU和Windows / NetWare网络操作系统，但也有一部分是采用UNIX系列操作系统的。
- 可以满足中小型网络用户的数据处理、文件共享、Internet接入及简单数据库应用的需求。

工作组级服务器较入门级服务器来说性能有所提高，功能有所增强，有一定的可扩展性，但容错和冗余性能仍不完善，也不能满足大型数据库系统的应用，但价格也比前者贵许多，一般相当于2~3台高性能的PC品牌机总价。

3）部门级服务器

这类服务器是属于中档服务器之列，一般都是支持双CPU以上的对称处理器结构，具备比较完全的硬件配置，如磁盘阵列、存储托架等。部门级服务器的最大特点就是，除了具有工作组服务器全部服务器特点外，还集成了大量的监测及管理电路，具有全面的服务器管理能力，可监测如温度、电压、风扇、机箱等状态参数，结合标准服务器管理软件，使管理人员及时了解服务

器的工作状况。同时，大多数部门级服务器具有优良的系统扩展性，能够满足用户在业务量迅速增大时及时在线升级系统，充分保护了用户的投资。它是企业网络中分散的各基层数据采集单位与最高层的数据中心保持顺利连通的必要环节，一般为中型企业的首选，也可用于金融、邮电等行业。

部门级服务器一般采用 IBM、SUN 和 HP 各自开发的 CPU 芯片，这类芯片一般是 RISC 结构，所采用的操作系统一般是 UNIX 系列操作系统，现在的 Linux 也在部门级服务器中得到了广泛应用。以前能生产部门级服务器的厂商通常只有 IBM、HP、SUN、COMPAQ（现在也已并入 HP）这么几家，不过现在随着其他一些服务器厂商开发技术的提高，现在能开发、生产部门级服务器的厂商比以前多了许多。国内也有好几家具备这个实力，如联想、曙光、浪潮等。

部门级服务器可连接 100 个左右的计算机用户，适用于对处理速度和系统可靠性高一些的中小型企业网络，其硬件配置相对较高，其可靠性比工作组级服务器要高一些，当然其价格也较高（通常为 5 台左右高性能 PC 机价格总和）。由于这类服务器需要安装比较多的部件，所以机箱通常较大，采用机柜式的。

4）企业级服务器

企业级服务器是属于高档服务器行列，正因如此，能生产这种服务器的企业也不是很多，但同样因没有行业标准硬件规定企业级服务器需达到什么水平，所以现在也看到了许多本不具备开发、生产企业级服务器水平的企业声称自己有了企业级服务器。企业级服务器最起码是采用 4 个以上 CPU 的对称处理器结构，有的高达几十个。另外一般还具有独立的双 PCI 通道和内存扩展板设计，具有高内存带宽、大容量热插拔硬盘和热插拔电源、超强的数据处理能力和群集性能等。这种企业级服务器的机箱就更大了，一般为机柜式的，有的还由几个机柜来组成，像大型机一样。企业级服务器产品除了具有部门级服务器全部服务器特性外，最大的特点就是它还具有高度的容错能力、优良的扩展性能、故障预报警功能、在线诊断和 RAM、PCI、CPU 等具有热插拔性能。

有的企业级服务器还引入了大型计算机的许多优良特性，如 IBM 和 SUN 公司的企业级服务器。这类服务器所采用的芯片也都是几大服务器开发、生产厂商自己开发的独有 CPU 芯片，所采用的操作系统一般也是 UNIX（Solaris）或 Linux。目前在全球范围内能生产高档企业级服务器的厂商也只有 IBM、HP、SUN 这么几家，绝大多数国内外厂家的企业级服务器都只能算是中、低档企业级服务器。企业级服务器适合运行在需要处理大量数据、高处理速度和对可靠性要求极高的金融、证券、交通、邮电、通信或大型企业。企业级服务器用于联网计算机在数百台以上、对处理速度和数据安全要求非常高的大型网络。企业级服务器的硬件配置最高，系统可靠性也最强。

## 4.4 Linux 安装及部署

### 4.4.1 注意事项

在安装部署之前，需要考虑以下几个安装时的选项。

#### 1. 是否安装 SELinux

SELinux（安全增强型 Linux）通过在 Fedora/RHEL 的内核实施强制访问控制策略来提高系统

的安全性。默认情况下，Fedora 安装 SELinux 的强制模式。如果你不打算使用 SELinux，你可以在系统安装时更改为自由模式。因为 SELinux 在文件上设置扩展属性，使 SELinux 系统上的安全属性关闭可能是一个耗时的过程。

2. 选择安装图形用户界面

在大多数系统中，除了用于服务器，你可能要安装一个图形用户界面（即桌面）。Fedora/RHEL 安装默认为 GNOME。

3. 选择安装软件和服务

当你想在系统上安装更多的软件包，更新和增加的软件包之间相互作用。监听网络连接的服务器软件包增加系统可以被攻击的方式，使系统更加脆弱。包括额外的服务，还可以减慢系统。

如果你想开发系统或用一个系统来学习，额外的软件包和服务可能是有用的。对于一个更安全的产品系统，最好是安装和维护所需的最低数量的软件包，使只有需要的服务运行。

### 4.4.2 其他需求

1. 硬盘空间

Fedora/RHEL 需要的硬盘空间取决于你安装哪一个版本的 Fedora/RHEL，安装哪些软件包，安装了多少种语言，和你的用户数据（你的文件）需要多大的空间。Fedora 操作系统通常需要 2～8GB，虽然一个最小系统低至 90MB RAM 也可以运行，但用户数据需要额外的空间。保留至少 5%的自由空间给文件系统进行操作。RHEL 系统的信息请参阅 www.redhat.com/rhel/compare。

2. BIOS 设置

现代计算机可以设置从一个 CD/DVD、硬盘或 USB 闪存驱动器引导。 BIOS（基本输入/输出系统）确定系统尝试引导每个设备的顺序。你可能需要改变这个顺序：请确保在 BIOS 设置尝试从 CD/DVD 启动之前，它会尝试从硬盘引导。

3. CMOS

CMOS 是永久的记忆存储硬件配置信息。要更改 BIOS 设置，你需要编辑存储在 CMOS 中的信息。当系统启动时，它会显示一个如何进入系统设置程序或 CMOS 设置模式的简短消息。通常你需要在引导系统时按 Del 或 F2 键，然后移动屏幕上的光标到处理引导系统的行。一般有三个或四个设备，系统将按照引导列表进行尝试，如果第一次尝试失败，系统将尝试第二个设备，并依此类推。操作列表以便使 CD/DVD 是第一选择，保存列表，并重新启动。更多信息请参阅硬件/BIOS 手册。

## 4.5 大规模自动部署安装 Linux

企业一般都会成批量地采购服务器，并在其上部署安装操作系统，无论 Linux 还是 Windows 都是如此。这是一个非常耗费时间、人力的过程，经常有系统管理员被这样重复、枯燥的生活折磨得痛苦不堪。对于一个 500 强的企业来说尤其如此，人力和时间是非常宝贵的资源，如何使得人力

和时间用在刀刃上呢？幸亏当今有了相应的技术来解决这个问题，下面让我们来看看在 500 强企业中都是如何来解决这个问题的。

### 4.5.1 PXE 技术

在如今，一般都会采用 PXE 技术来解决大规模自动部署和安装 Linux 的问题。PXE（Preboot Execute Environment）是由 Intel 公司开发的最新技术，基于 Client/Server 的网络模式，支持工作站通过网络从远端服务器下载映像，并由此支持来自网络的操作系统的启动过程。在其启动过程中，终端要求服务器分配 IP 地址，再用 TFTP（Trivial File Transfer Protocol）或 MTFTP（Multicast Trivial File Transfer Protocol）协议下载一个启动软件包到本机内存中并执行，由这个启动软件包完成终端基本软件设置，从而引导预先安装在服务器中的终端操作系统。PXE 可以引导多种操作系统，如：Windows 95/98/2000/XP/2003/Vista/2008、Linux 等。

PXE 最直接的表现是，在网络环境下工作站可以省去硬盘，但又不是通常所说的无盘站的概念，因为使用该技术的 PC 在网络方式下的运行速度要比有盘 PC 快 3 倍以上。当然使用 PXE 的 PC 也不是传统意义上的 Terminal 终端，因为使用了 PXE 的 PC 并不消耗服务器的 CPU、RAM 等资源，故服务器的硬件要求极低。

网络克隆 PXE 现在最为广泛的一个应用是网吧的无盘技术。在有盘领域的网络维护和安装中，PXE 可以是最好用的网吧系统统一安装和维护的引导技术，PXE 的引导速度和稳定性都是一流的。

### 4.5.2 搭建 Yum 源

为了完成后续的大规模操作，需要先搭建一个 Yum 源，用于安装自动部署所需要的 DHCP、TFTP 和 VSFTP。

Yum 是 Yellowdog Updater Modified 的缩写。Yellowdog 是一个 Linux 的 distribution，RH 将这种升级技术利用到自己的 distribution 形成了现在的 Yum，感觉上 Yum 和 APT 的原理类似，但是 APT 是编译代码，执行效率远高于使用 Python 写成的 Yum。这是 Yum 的主页。Yum 的理念是使用一个中心仓库（repository）管理一部分甚至一个 distribution 的应用程序相互关系，根据计算出来的软件依赖关系进行相关的升级、安装、删除等操作，减少了 Linux 用户一直头痛的 dependencies 的问题。

具体的搭建步骤如下。

（1）挂载 Linux 光盘：

```
#mount /dev/hdc  /mnt
```

（2）编辑/etc/yum.conf，增加下面的内容即可：

```
[cdrom]
name=cdrom
baseurl=file:///mnt/Server
enabled=1
gpgcheck=1
```

## 4.5.3 安装相关服务

### 1. 安装VSFTP服务

使用如下命令安装VSFTP服务：

```
# yum install vsftpd* -y
# chkconfig vsftpd on
# service vsftpd restart
```

### 2. 复制PXE启动时需要的文件资料

（1）复制必要的文件。
使用如下命令：

```
# cp /usr/lib/syslinux/pxelinux.0 /tftpboot/
# mkdir /tftpboot/pxelinux.cfg
# cp /cdrom/isolinux/isolinux.cfg /tftpboot/pxelinux.cfg/default
# cp /cdrom/images/pxeboot/initrd.img /tftpboot/
# cp /cdrom/images/pxeboot/vmlinuz /tftpboot/
```

（2）修改/tftpboot/pxelinux.cfg/default文件。
使用如下命令：

```
# chmod u+w /tftpboot/pxelinux.cfg/default //文件默认只读
# vim /tftpboot/pxelinux.cfg/default
default linux
prompt 1
timeout 6 //超时时间，默认为600，可以不改。
display boot.msg
F1 boot.msg
F2 options.msg
F3 general.msg
F4 param.msg
F5 rescue.msg
label linux
kernel vmlinuz
append initrd=initrd.img ks=ftp://192.168.1.8/ks.cfg //ks.cfg是kickstart安
装配置文件，系统就是按照ks.cfg来安装的。
```

### 3. 安装配置DHCP服务

使用以下命令安装DHCP服务：

```
# yum install dhcp.i386 dhcp-devel.i386 -y
```

然后，生成主配置文件 dhcpd.conf：

```
# cp /usr/share/doc/dhcp-3.0.5/dhcpd.conf.sample /etc/dhcpd.conf
```

接着，修改主配置文件 dhcpd.conf，修改如下配置：

```
# vim /etc/dhcpd.conf
ddns-update-style interim;
ignore client-updates;
subnet 192.168.1.0 netmask 255.255.255.0 {      //所属网段及掩码
# --- default gateway
option routers 192.168.1.8;                     //路由器 IP，可以写网关 IP
option subnet-mask 255.255.255.0;
filename "pxelinux.0";                          //PXE 得到 IP 以后的引导文件
next-server 192.168.1.8;                        //服务器 IP 地址
# option nis-domain "domain.org";               //注销
# option domain-name "domain.org";              //注销
option domain-name-servers 192.168.1.8;         //DNS 服务器 IP
option time-offset -18000; # Eastern Standard Time
# option netbios-node-type 2;
range dynamic-bootp 192.168.1.100 192.168.1.200; //IP 地址池范围
default-lease-time 21600;
```

最后，启动 DHCP 服务：

```
# service dhcpd restart
```

### 4．生成 kickstart 配置文件

按照以下步骤进行配置。

（1）安装 kickstart 包。

```
# yum list *kic*
Loaded plugins: rhnplugin, security
This system is not registered with RHN.
RHN support will be disabled.
Available Packages
pykickstart.noarch 0.43.3-1.el5 cdrom
system-config-kickstart.noarch 2.6.19.8-2.el5 cdrom
# yum install system-config-kickstart.noarch
```

（2）生成 ks.cfg 安装配置文件。

① 图形化界面配置：（终端中运行 system-config-kickstart 命令），如图 4-9～图 4-13 所示。




图 4-9　“基本配置”中要修改的项目




图 4-10　“安装方法”中要修改的项目




图 4-11　“分区信息”创建目标工作站的分区表




图 4-12　“网络配置”（单击“添加网络设备”按钮，最后单击“确定”按钮）

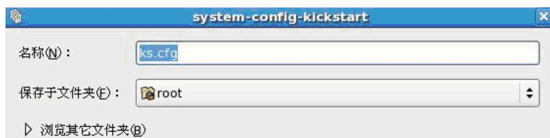



图 4-13　选择“文件”中的“保存文件”命令，最后单击“保存“按钮

② 修改新生成的 ks.cfg 文件# cat /root/anaconda-ks.cfg //安装系统时自动记录的安装信息，然后选择复制%packages 后面的所有记录系统安装时的软件包的安装情况的配置到 ks.cfg 文件后面。

（3）将配置文件复制到指定位置，我们在/tftpboot/pxelinux.cfg/default 文件中曾设置 ks=ftp://192.168.1.8/ks.cfg。

将文件复制到该位置：

```
# cp ks.cfg /var/ftp/ // /var/ftp 目录是 VSFTP 服务的根目录
```

## 5. 复制 Linux 系统安装文件

在配置 ks.cfg 文件时，我们设置的路径为/pub，如图 4-14 所示。




图 4-14　设置路径

并使用如下命令进行加载：

```
# umount /dev/hdc
# mount /dev/hdc /var/ftp/pub/
```

### 6. 确定相关服务的状态

使用如下命令，关闭 IPTables 防火墙和 SELinux 安全机制：

```
# service iptables stop      //清除防火墙规则
# setenforce 0               //关闭 selinux
```

开启 TFTP、DHCP 和 VSFTP 服务：

```
# chkconfig tftp on          //TFTP 服务开启命令
# chkconfig dhcpd on
# chkconfig vsftpd on
# service xinetd restart     //TFTP 服务属于它的子服务
# service dhcpd restart
# service vsftpd restart
```

通过以上配置后，即可跳过 DHCP 而通过 TFTP 开始启动计算机，如图 4-15 所示，则可以大规模地自动部署安装 Linux 了。

```
CLIENT IP: 192.168.1.200  MASK: 255.255.255.0  DHCP IP: 192.168.1.8
GATEWAY IP: 192.168.1.8

PXELINUX 3.11 2005-09-02  Copyright (C) 1994-2005 H. Peter Anvin
UNDI data segment at:   0009C7F0
UNDI data segment size: 24D0
UNDI code segment at:   0009ECC0
UNDI code segment size: 0A0D
PXE entry point found (we hope) at 9ECC:0106
My IP address seems to be C0A801C8 192.168.1.200
ip=192.168.1.200:192.168.1.8:192.168.1.8:255.255.255.0
TFTP prefix:
Trying to load: pxelinux.cfg/01-00-0c-29-6e-ac-e6
Trying to load: pxelinux.cfg/C0A801C8
Trying to load: pxelinux.cfg/C0A801C
Trying to load: pxelinux.cfg/C0A801
Trying to load: pxelinux.cfg/C0A80
Trying to load: pxelinux.cfg/C0A8
Trying to load: pxelinux.cfg/C0A
Trying to load: pxelinux.cfg/C0
Trying to load: pxelinux.cfg/C
Trying to load: pxelinux.cfg/default
boot:
```

图 4-15　成功启动

# 第三篇
# 企业 Linux 安全运维实战

## 战略上藐视敌人，战术上重视敌人

“战略上藐视敌人，战术上重视敌人”，是伟大领袖毛泽东主席的战略战术思想，是克敌制胜不可分割的两个方面。所谓战略是比喻决定全局的策略；战术是比喻解决局部问题的方法；所谓敌人指真正的敌人，也可以指工作中遇到的困难和障碍。

在安全运维工作中，可以以之为指导思想，要把困难想在前面，做好周密的布署，要不畏艰难，从战略上藐视困难，而从战术上重视困难，做好相关的准备工作。

# 第5章

# 高屋建瓴：“四步”完成企业Linux系统安全防护

## 本章导读

企业级Linux作为500强企业信赖的操作系统平台，其自身的运行安全（第3章所讲述的企业级Linux安全设计的内容）值得重视。而根据操作系统的核心概念以及500强企业运维人员以及安全管理人员的多年运维经验，将企业级Linux系统的安全防护可以准确定位到以下四个方面：

- Linux文件系统访问安全
- Linux进程安全
- Linux用户管理安全
- Linux日志管理安全

因此，本章将从这个四个方面出发，来介绍该500强企业对于这四个方面的安全防护技术、工具和策略。

## 5.1 分析：企业 Linux 系统安全威胁

在实际工作中，该 500 强企业的信息安全首席官（CSO，Chief Security Officer）和 Linux 管理员需要成立专门的工作小组，对企业级 Linux 系统可能发生的安全问题进行分析、总结和归纳，以此来确定企业级 Linux 系统安全防护的内容、步骤和方法，才能在后面的工作中进行具体实施。

根据企业级 Linux 系统的文件系统、进程、用户管理以及日志管理等方面的分析以及可能存在的安全威胁，可以总结得出如表 5-1 所示的企业级 Linux 系统安全威胁列表。

表 5-1　企业级 Linux 系统安全威胁列表

| | 解决的主要安全威胁 | 安全威胁牵涉到的人员及操作 |
|---|---|---|
| 文件系统防护 | 避免有意/无意的文件篡改、越权访问、根用户（root）权限滥用 | 企业内部用户误操作、外部黑客的恶意删除 |
| 进程安全防护 | 避免非法进程运行、正常进程非法终止 | 外部黑客的后门注入及非法进程运行 |
| 用户安全管理 | 避免弱密码被攻破 | 外部黑客使用暴力破解等方式获取密码 |
| 日志管理 | 通过对企业内部用户及外部黑客在系统中行动产生相应的日志记录进行审计，发现安全问题和蛛丝马迹 | 用户、黑客在系统中的行动产生日志记录 |

## 5.2 理念：企业级 Linux 系统安全立体式防范体系

Linux 的系统安全防护细分为以下四个关键部分（参见图 5-1）。

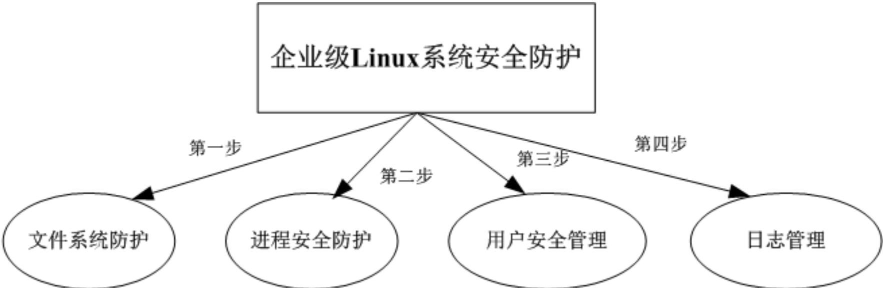



图 5-1　企业级 Linux 系统安全立体式防范体系

- 文件系统保护。Linux 系统的最大特点就是文件系统，其中所有的设备都是通过文件进行操作和管理。
- 进程安全保护。进程是所有现代操作系统的一个核心概念，很多攻击的发动、黑客的入侵都是通过进程来实现的。
- 用户安全管理。Linux 作为一种多任务、多用户的操作系统，在同一时间段上可能为众多用户使用，且用户的管理直接关系到整个系统的安全。
- 日志管理。从 Ext 到 Ext2，从 Ext2 再到 Ext3，乃至以后的 Ext4 或者更高版本，Linux

系统历来以强大、丰富和完整的日志系统著称。通过管理日志，可以清晰地了解系统的运行状况，也能从各种蛛丝马迹中发现入侵和快速地阻止入侵。

下面将会针对以上四点给出详细的安全防护解决方案，并会灵活机动地使用一些高效和成熟的安全工具和手段来辅助进行安全防护。

最后，本章还会介绍一款 Linux 下非常有名的主机入侵检测系统软件 LIDS。企业系统管理员可以应用该软件来实时、方便地检测和阻断有可能发生的入侵行为，该工具可以说是上述立体式防范体系的一个有益的补充。

## 5.3 企业 Linux 文件系统安全防护

### 5.3.1 企业 Linux 文件系统的重要文件及目录

要保护文件先要对被保护的文件进行分类，做到有的放矢。对 Linux（以红帽企业版为蓝本）中的文件进行了解和分类，确定如下的一些重要文件及其目录。

- /：根目录。在 Windows、DOS 或者其他类似的操作系统里面，每个分区都会有一个相应的根目录。但是 Linux 和其他 UNIX 系统则把所有的文件都放在一个目录树里面，/就是唯一的根目录。一般来讲，根目录下面很少保存什么文件，或者只有一个内核映像在这里。
- /boot：很多 Linux 系统把内核映像和其他一些和启动有关的文件都放在这里。
- /tmp：一般只有启动时产生的临时文件才会放在这个地方。我们自己的临时文件都放在/var/tmp。
- /mnt：这个目录下面放着一些用来安装其他设备的子目录，比如说/mnt/cdrom 或者/mnt/floppy。在有些 Linux 中这个目录是被/mount 代替的。
- /lib：启动的时候所要用到的库文件都放在这个目录下。那些非启动用的库文件都会放在/usr/lib 下。内核模块是被放在/lib/modules/（内核版本）下的。
- /proc：这个目录在磁盘上其实是不存在的。里面的文件都是关于当前系统的状态，包括正在运行的进程、硬件状态、内存使用的多少等。
  - /proc/cpuinfo：关于处理器的信息，如类型、厂家、型号和性能等。
  - /proc/devices：当前运行内核所配置的所有设备清单。
  - /proc/dma：当前正在使用的 DMA 通道。
  - /proc/filesystems：当前运行内核所配置的文件系统。
  - /proc/interrupts：正在使用的中断，和曾经有多少个中断。
  - /proc/ioports：当前正在使用的 I/O 端口。
- /dev：这个目录下保存着所有的设备文件。里面有一些由 Linux 内核创建的用来控制硬件设备的特殊文件。
- /var：这里有一些被系统改变过的数据。比如说/var/tmp，就是用来存储临时文件的。还有很多其他的进程和模块把它们的记录文件也放在这个地方，包括以下一些重要的子目录。

  - /var/log：这里存放着绝大部分的记录文件。随着时间的增长，这个目录会变得很庞大，所以要定期清理。
  - /var/run：包括了各种运行时的信息。
  - /var/lib：包括了一些系统运行时需要的文件。
  - /var/spool：邮件、新闻、打印序列的所在地。
- /root：root 用户的主目录。
- /home：一般用户的主目录都会放在这个目录下。在 Linux 下，可以通过#cd～来进入自己的主目录。
- /etc：这里保存着绝大部分的系统配置文件。相对来讲，单个用户的系统配置文件会保存在这个用户自己的主目录里面。下面列举其中一些重要的文件和子目录。
  - /etc/group：组用户信息。
  - /etc/passwd：包含所有的用户信息，如密码，登录 shell 等。
  - /etc/fstab：配置系统有哪些文件系统。
  - /etc/inittab：配置 init 在不同运行级别下分别如何启动系统。
  - /etc/X11：这里放着 X 窗口系统（Linux 中的图形用户界面系统）所需要的配置文件。XF86Config 就是把配置存储到这个地方的。/etc/X11/fonts 里面放着一些服务器需要的字体，还存放一些窗口管理器存放的配置文。
  - /etc/init.d：这个目录保存着启动描述文件，包括各种模块和服务的加载描述。所以如果不清楚的话，千万不要随便删除其中的文件，这里存放的文件都是系统自动进行配置的，不需要用户配置。
  - /etc/rcS.d：这里放着一些连接到/etc/init.d 的文件，根据 runlevel 的不同而执行相应的描述。这里的文件名都是由 S 来开头的，然后是一个两位的数字——表示各种服务启动的顺序。比如，S24foo 就是在 S42bar 前面执行的。接着就是相应的连接到/etc/init.d 下面的文件的名字了。
  - /etc/rc0.d～/etc/rc6.d：这里面也是一些连接文件，和/etc/rcS.d 差不多。不同的是，这些只会在指定的 runlevel 下运行相应的描述。0 表示关机，6 表示重启。所有以 K 开头的文件表示关闭，所有以 S 开头的文件表示重启。目前来讲，文件的命名方式和/etc/rcS.d 是一样的。
- /bin、/sbin：这里分别放着启动时所需要的普通程序和系统程序。很多程序在启动以后也很有用，它们放在这个目录下是因为它们经常被其他程序调用。
- /usr：这是一个很复杂、庞大的目录。除了上述目录之外，几乎所有的文件都存放在这里。下面列举其中一些重要的子目录。
  - /usr/X11R6、/usr/X11、/usr/Xfree86：这里保存着 X 窗口系统所需要的文件，它的目录结构和/usr 是一样的。
  - /usr/bin：二进制可执行文件存放的目录，这里存放着绝大部分的应用程序。
  - /usr/sbin：这里存放着绝大部分的系统程序。
  - /usr/games：游戏程序和相应的数据会存放在这里。
  - /usr/include：这个目录保存着 C 和 C++的头文件。
  - /usr/lib：启动时用不到的库文件都会存放在这里。
  - /usr/info：这里保存着 GNU Info 程序所需要的数据。

- /usr/man：这里保存着 man 程序所需要的数据。
- /usr/src：这里保存着源代码文件。
- /usr/doc：这里保存着各种文档文件。这些文件可以帮助你了解 Linux、解决问题和提供一些技巧。
- /usr/local：这里面保存着本地计算机所需要的文件。在用户进行远程访问的时候特别有意义。这个目录在有些 Linux 系统下就是一个单独的分区，存放一些这台机子所属的那个用户的文件。里面的结构和/usr 是一样的。
- /usr/shared、/usr/share：这里保存着各种共享文件。

### 5.3.2 文件/目录访问权限

Linux 系统中的每个文件和目录都有访问许可权限，通过其确定谁可以通过何种方式对文件和目录进行访问和操作。文件或目录的访问权限分为只读、只写和可执行三种。以文件为例，只读权限表示只允许读其内容，而禁止对其做任何的更改操作；只写权限允许对文件进行任何的修改操作；可执行权限表示允许将该文件作为一个程序执行。文件被创建时，文件所有者自动拥有对该文件的读、写和可执行权限，以便于对文件的阅读和修改。用户也可根据需要将访问权限设置为需要的任何组合。

有三种不同类型的用户可对文件或目录进行访问：文件所有者、同组用户和其他用户。所有者一般是文件的创建者，他可以允许同组用户有权访问文件，还可以将文件的访问权限赋予系统中的其他用户。在这种情况下，系统中的每一位用户都能访问该用户拥有的文件或目录。

每一个文件或目录的访问权限都有三组，每组用三位表示，分别为文件属主的读、写和执行权限；与属主同组的用户的读、写和执行权限；系统中其他用户的读、写和执行权限。当用 ls -l 命令显示文件或目录的详细信息时，最左边的一列为文件的访问权限。例如：

```
# ls -l
总计 76
-rw------- 1 root root   797 11-06 20:41 anaconda-ks.cfg
drwxr-xr-x 2 root root  4096 11-06 13:50 Desktop
-rw-r--r-- 1 root root 44843 11-06 20:40 install.log
```

-rw-r--r-- 1 root root 44843 11-06 20:40 install.log 中横线代表空许可（即表示不具有该权限）。r 代表只读，w 代表写，x 代表可执行。注意：这里共有 10 个位置。第 1 个字符指定了文件类型。在通常意义上，一个目录也是一个文件。如果第 1 个字符是横线，表示是一个非目录的文件。如果是 d，表示是一个目录。后面的 9 个字符每三个构成一组，依次表示文件属主、组用户、其他用户对该文件的访问权限。

例如：

```
-rw-r--r--   install.log
```

表示文件 install.log 的访问权限，说明 install.log 是一个普通文件；install.log 的属主有读写权限；与 install.log 属主同组的用户只有读权限；其他用户也只有读权限。

确定了一个文件的访问权限后，用户可以利用 Linux 系统提供的 chmod 命令来重新设定不同的访问权限，也可以利用 chown 命令来更改某个文件或目录的所有者。

### 5.3.3 字母文件权限设定法

字母设定法的一般使用形式为：chmod [who] [+ | - | =] [mode] 文件名。

其中，操作对象 who 可以是下述字母中的任意一个或者为各字母的组合。

- u 表示“用户（user）”，即文件或目录的所有者。
- g 表示“同组（group）用户”，即与文件属主有相同组 ID 的所有用户。
- o 表示“其他（others）用户”。
- a 表示“所有（all）用户”。其为系统默认值。

操作符号如下。

- +：添加某个权限。
- –：取消某个权限。
- =：赋予给定权限并取消其他所有权限（如果有的话）。

设置 mode 所表示的权限可用下述字母的任意组合。

- r：可读。
- w：可写。
- x：可执行。只有目标文件对某些用户是可执行的或该目标文件是目录时才追加 x 属性。
- s：在文件执行时把进程的属主或组 ID 置为该文件的文件属主。方式“u+s”设置文件的用户 ID 位，“g+s”设置组 ID 位。
- t：将程序的文本保存到交换设备上。
- u：与文件属主拥有一样的权限。
- g：与文件属主同组的用户拥有一样的权限。
- o：与其他用户拥有一样的权限。

### 5.3.4 数字文件权限设定法

数字设定法是与文字设定法功能等价的设定方法，只不过比文字设定法更加简便。数字表示的属性的含义为：0 表示没有权限，1 表示可执行权限，2 表示可写权限，4 表示可读权限，然后将其相加。所以数字属性的格式应为 3 个从 0～7 的八进制数，其顺序是（u）、（g）、（o）。其他的与文字设定法基本一致。

如果想让某个文件的属主有“读/写”两种权限，需要把 4（可读）+2（可写）=6（读/写）。

数字设定法的一般形式为：chmod [mode]文件名。

下面给出一些使用该数字设定法的例子。

（1）设定文件 install.txt 的属性为：文件属主（u）拥有读、写权限；与文件属主同组的用户（g）拥有读权限；其他人（o）拥有读权限。

```
#chmod 644 install.txt
```

（2）设定 example.c 文件的属性为：文件主本人（u）具有可读、可写、可执行权限；与文件主同组的人（g）具有可读、可执行权限；其他人（o）没有任何权限。

```
#chmod 750 example.c
```

### 5.3.5 特殊访问模式及粘贴位的设定法

通过前面的介绍我们知道，Linux 系统中的每一个文件都有一个所有者，表示该文件是谁创建的。同时，该文件还有一个组编号，表示该文件所属的组，一般为文件所有者所属的组。并且，在一般情况下，我们也可以通过设定对文件的权限来控制对其的相关操作。

在此情况中，如果是一个可执行文件，那么在执行时，一般该文件只拥有调用该文件的用户具有的权限。而 setuid、setgid 则可以来改变这种设置。

- setuid：设置使文件在执行阶段具有文件所有者的权限。典型的文件是/usr/bin/passwd。如果一般用户执行该文件，则在执行过程中，该文件可以获得 root 权限，从而可以更改用户的密码。
- setgid：该权限只对目录有效。目录被设置该位后，任何用户在此目录下创建的文件都具有和该目录所属的组相同的组。
- sticky bit：该位可以理解为防删除位，也称为粘贴位。一个文件是否可以被某用户删除，主要取决于该文件所属的组是否对该用户具有写权限。如果没有写权限，则这个目录下的所有文件都不能被删除，同时也不能添加新的文件。如果希望用户能够添加文件但同时不能删除文件。则可以对文件使用 sticky bit 位。设置该位后，即便用户对目录具有写权限，也不能删除该文件。

在前面我们讨论了通过文字设定和数字设定的方法来设定文件、目录的访问权限问题，同样的道理，下面我们通过这两种方法来介绍如何操作这些标志。操作这些标志与操作文件权限的命令是一样的，都是使用 chmod 命令来进行。

#### 1．文字设定法

chmod u+s filename：为文件 filename 加上 setuid 标志。

chmod g+s dirname：为目录 dirname 加上 setgid 标志。

chmod o+t filename：为文件 filename 加上 sticky 标志。

#### 2．数字设定法

对一般文件通过三组八进制数字来置标志，如 444、777、644 等。如果设置这些特殊标志，则在这组数字之前另加一组八进制数字。如 4666、4777 等。这一组八进制数字三位的意义如下。

- setuid 位：如果该位为 1，显示为"S"，则表示设置 setuid，其显示在原来的 x 标志位上。
- setgid 位：如果该位为 1，显示为"S"，则表示设置 setgid，其显示在原来的 x 标志位上。
- sticky 位：如果该位为 1，显示为"T"，则表示设置 sticky，其显示在原来的 x 标志位上。

设置完这些标志后，可以用 ls–l 命令来查看。如果有这些标志，则会在原来的执行标志位置上显示。如下所示。

r-srw-r--：表示有 setuid 标志。

rwxrwsrw-：表示有 setgid 标志。

rwxrw-rwt：表示有 sticky 标志。

如果本来在该位上有 x，则这些特殊标志显示为小写字母（s，s，t）。否则，显示为大写字母（S，S，T）。

另外非常值得读者注意的是，虽然 setuid、setgid 机制非常方便实用，但是由于提升了执行者的权限，因而不可避免地存在许多安全隐患和风险，所以作者并不太赞成广大读者广泛使用，并且，在实际的系统管理过程中，我们还经常需要找出设置有这些标志的文件，并对它们进行检查和清理。一般我们可以使用如下命令来对系统中的具有特殊标志的文件进行寻找：

```
#find -perm +6000 -type f -exec ls -ld {}\; > setuid.txt&
```

## 5.3.6 使用文件系统一致性检查工具：Tripwire

### 1. Tripwire 工具简介

系统的正常运行要靠系统程序的正常运转，而程序的运行又与其可执行文件息息相关。所以，维护系统完整性是确保系统安全的一项基本工作。我们这里的系统完整性是指系统中核心配置文件（包括密码文件等）和可执行文件的完整性，也就是说系统中的核心文件和程序文件没被非法修改。

一方面，如果可执行文件被恶意修改的话，如改变、插入或删除等，将直接威胁到系统的安全性。大多数情况下，黑客渗入到系统后会立即修改某些系统文件以创建后门，如用准备好的替代物换掉系统中原有的/bin/login 文件以便使其不用口令便能登录系统；然后再修改某些文件，例如/bin/ls 等，以便隐藏其行径。如果我们没能发现这些改变的话，那无异于身处险境却还以为很安全，这就为黑客的长期入侵提供了非常有利的条件，同时也意味着我们的损失将更大。另一方面，如果用户的核心账户文件被修改的话，就有可能造成正常用户无法登录或者是非法用户的潜入等。为了改变这种被动的局面，我们需要一种文件完整性检查工具，使得当系统文件被恶意修改后能及时发现，从而为进一步处理创造条件。

Tripwire 是 UNIX 安全规范中最有用的工具之一，l992 年由 Eugene Spafford 和 Gene Kim 在 Purdue 大学进行开发的，其软件包在 1998 年被商业化，现在由 Tripwire Security Inc.来维护。该公司支持软件包的商业和免费两种版本，能在多数 UNIX 系统中运行。

由于使用了多达四种的 Hash 算法，因此准确度非常高。Tripwire 可检测多达十多种的 UNIX 文件系统属性和二十多种的 NT 文件系统（包括注册表）属性。Tripwire 首先使用特定的特征码函数为需要监视的系统文件和目录建立一个特征数据库，所谓特征码函数就是使用任意的文件作为输入，产生一个固定大小的数据（特征码）的函数。入侵者如果对文件进行了修改，即使文件大小不变，也会破坏文件的特征码。利用这个数据库，Tripwire 可以很容易地发现系统的丝毫细微的变化。而且文件的特征码几乎是不可能伪造的，系统的任何变化都逃不过 Tripwire 的监视。

Tripwire 作为一种优秀的安全工具，其具有：自保护性、平台间可移植性、可配置性、可扩展性和自包容性的特点。

与大多数完整性检查程序相同，对于需要监视的文件，Tripwire 会使用校验和来为文件的某个状态生成唯一的标识（又称为“快照”），并将其存放起来以备后用。当 Tripwire 程序运行时，它先计算新的标识，并与存放的原标识加以比较，如果发现不匹配的话，它就报告系统管理人员文件已经被修改。接下来，系统管理员就可以利用这个不匹配来判断系统是否遭到了入侵。例如，如果 Tripwire 已经为/bin/login 和/bin/ls 存放了快照，那么对它们的尺寸、inode 号、权限以及其他属性的任何修改，都逃不过 Tripwire 的详细和周密的检查。尤其是对于文件内容的修改，即使只改变了一个字节，Tripwire 也能察觉得到，因为校验和是针对文件整体的。

该软件基于预编写的策略工作，在基准数据库生成时，会根据策略文件中的规则读取指定的

文件，同时生成该文件的数字签名并存储在 Tripwire 自己的数据库中。为了达到最大限度的安全性，Tripwire 提供了四种 Hash 算法——CRC32、MD5、SHA、HAVAL 来生成签名。通常情况下采用前两种算法生成签名就足够了，当然也可全部采用。不过后两种算法对系统资源的耗费较大，使用时可根据文件的重要性作灵活的取舍。

进行完整性检查时，Tripwire 会根据策略文件中的规则对指定的文件重新生成一次数字签名，并将此签名与存储在数据库中的签名做对照。如果完全匹配，则说明文件没有被更改；如果不匹配，说明文件被改动了。然后在 Tripwire 生成的报告中查阅文件被改动的具体情况，如图 5-2 所示。

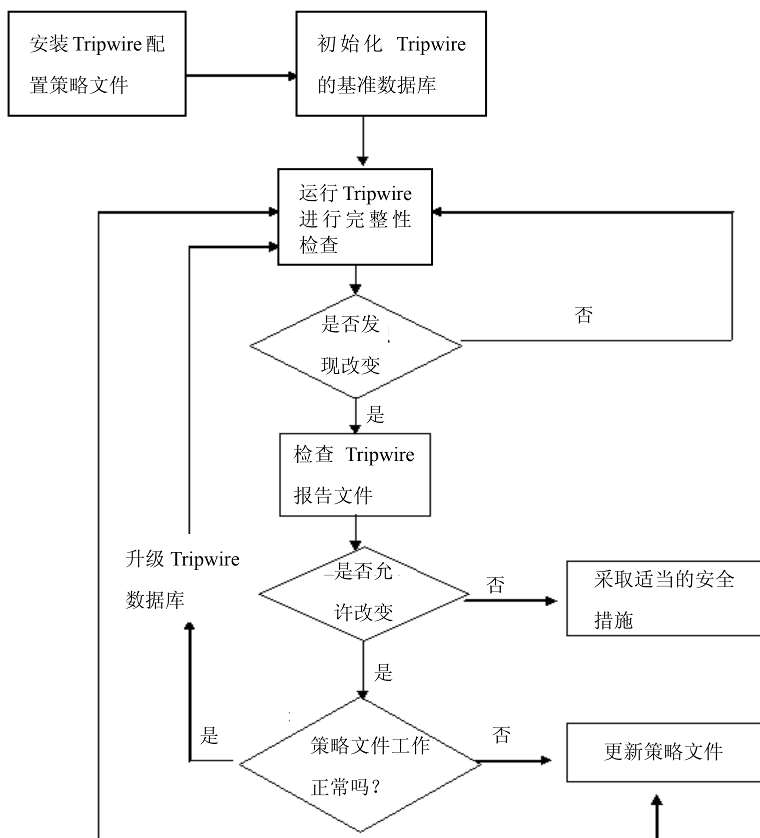



图 5-2　Tripwire 工作原理示意图

由上述可知，Tripwire 自身的基准数据库是非常重要的。如果基准数据库变得不可靠，那么完整性检查就没有意义。Tripwire 软件安装完毕后，已经对配置文件、策略文件以及数据库文件均进行了高强度的加密，同时默认策略中也对自身文件进行了完整性检查。当然，使用者自己也要做好这些文件的备份，因为 Tripwire 软件是不能恢复受损文件的，它只能详细列出每一个受损文件的情况。

Tripwire 在 UNIX 系统中的属性监控如下。

- 文件增加、删除、修改。
- 文件访问许可属性。
- iNode 及 Link 数量。
- Uid 及 Gid。
- 文件类型和大小。
- iNode 存储的磁盘设备号。
- iNode 指向的设备的设备号（适用于设备文件）。
- 分配的区块。
- 修改时间戳。

iNode 创建和修改的时间戳。

- 访问时间戳。
- 增长的文件。
- 缩小的文件。
- Hash 检查。

Tripwire 有四种操作模式：数据库生成、完整性检查、数据库更新和交互式更新。

- 数据库生成模式产生一个基准数据库，为未来的比较打下基础。
- 完整性检查是 Tripwire 的主要模式，把当前文件签名和基准数据库进行比较来进行检查。
- 另外两种更新模式允许用户调整 Tripwire 数据库以消除不感兴趣的结果以及应付正常的系统变化。例如当用户账号正常增加或删除时，不希望 Tripwire 重复报告/etc/passwd 文件被改动了。

#### 2. 选用 Tripwire 的四个核心散列函数

如上所述，该软件能够全面、精确地扫描出文件的任何微小的变化，与它所采用的多种高强度的散列算法是离不开的。散列（Hash）函数也称消息摘要（Message Digest）、哈希函数或杂凑函数等，其输入为一可变长 x，返回一固定长度串，该串 h 被称为输入 x 的 Hash 值（消息摘要），记作 h=H(x)。

Hash 函数 H 一般满足以下几个基本要求。

- 输入 x 可以为任意长度。单向散列函数能够处理任意长度的明文（至少是在实际应用中可能碰到的长度的明文），其生成的消息摘要数据块长度具有固定的大小，而且对同一个消息反复执行该函数总是得到相同的信息摘要。
- 输出数据串长度固定。即使使用的算法不同，但所有的散列值都拥有很多相同的属性。散列值的长度由算法的类型决定，与被散列的消息大小无关，一般为 128 或 160 比特长度。
- 易计算，给定任何 x，容易算出 H(x)。
- 单向函数，即给出一 Hash 值 h，很难反向计算出一特定输入 x，使 h=H(x)。所有的散列算法都是单向的，也就是说不能从散列值来获取原始消息，即使是原始消息的很少一部分信息都不可能获得。所以，有时候对一个数据的散列也叫做该数据的摘要，它可以作为一个消息的惟一标识，来保护消息的完整性。发送者发送数据的同时，也把该数据的散列值发送过去。接收者接收到消息以后，首先根据接收到的数据计算其散列值，然后和发送过来的散列值进行比较，就可以判断数据是否在发送过程中被修改。
- 惟一性，又叫冲突性，可分为弱冲突和强冲突两种。弱冲突是指给出一消息 x，找出一消息 y 同 x 相似且 H(x)=H(y)是计算不可行的，而强冲突是指找出任意两条消息 x、y，使 H(x)=H(y)也是计算不可行的。即使两个消息差别很小，如差别一两个比特，其散列值也完全不同。利用现在的算法技术，几乎不可能找到两个拥有相同散列值的不同消息。用同一个算法对某一消息进行散列计算只能获取惟一确定的散列值。
- 单向散列函数生成的信息摘要是不可预见的，消息摘要看起来和原始的数据没有任何的关系。而且，原始数据的任何微小变化都会对生成的信息摘要产生很大的影响。

1）Snefru 算法

Snefru 是 Ralph Merkle 设计的一种单向散列函数，它将任意长度的消息散列成 128 或者 256

位的值。

其算法首先将消息分成 512-m 的分组（m 是散列值的长度）。若输出是 128-位散列值，则每分组 384 位长；若输出是 255-位散列值，则每分组 256 位长。

算法的核心是函数 H，它将 512-位值散列成 m-位值。H 输出的前 m 位是这一分组的散列，余下的则舍弃掉。下一分组附在上一分组的散列后面，然后又进行散列（初始分组附在一串 0 之后）。在最后一个分组散列之后，将最先的 m 位附在消息长度的二进制表示之后并进行最后一次散列。

函数 H 基于另一个作用于 512-位分组的可逆分组密码函数 E。H 是 E 输出的最后 m 位与 E 输入的最先 m 位相异或的结果。

Snefru 的安全性取决于函数 E，它用几轮运算使数据随机化。每轮由 64 个随机化的子轮组成。在每一个子轮中用不同的字节作为 S-盒的输入，S-盒输出的一个字与消息相邻的两个字相异或。

2）HAVAL 算法

HAVAL 是一种长度可变的单向散列函数，它是 MD5 的改进版本。HAVAL 以 1025-位分组处理消息，是 MD5 的两倍。它有八个 32-位链接变量，也是 MD5 的两倍。它的轮数可在 3～5 轮中变化，其中每一轮为 16 步，并且它能产生长度为 128、160、192、224 或者 256 位的散列值。

HAVAL 是用高非线性的 7-变量函数取代了 MD5 的简单非线性函数，而且每一个函数都能够满足严格的雪崩规则的要求。每轮采用单个函数，但在每一步对输入进行了不同的置换。该算法有一个新的消息次序，并且每一步（第一轮除外）使用了不同的加法常数。该算法有两种环移方式。

该算法的核心为：

TEMP =（f（j，A，B，C，D，E，F，G）<<< 7）+（H <<< 11）+ M[i][r（j）] + K（j）

H = G；G = F；F = E；E = D；D = C；C = B；B = A；A = TEMP

其中，可变的轮数和可变的输出长度意味着该算法有 15 种不同的形式。

3）MD5 算法

MD5 算法是 MD4 的改进版本，它比 MD4 更为复杂，但是设计思想相似，并且也是产生 128-位散列。在此，我们就对该算法的实现流程做一个详细的分析。

（1）MD5 算法是对输入的数据进行补位，使得数据位长度 LEN 对 512 求余的结果是 448，即数据扩展至（K*512+448）位，也即（K*64+56）个字节，K 为整数。具体补位操作：补一个 1，然后补 0，直至满足上述要求。

（2）补数据长度：用一个 64 位的数字表示数据的原始长度 B，将 B 用两个 32 位数表示。这时，数据就被填补成长度为 512 位的倍数。

（3）初始化 MD5 参数：四个 32 位整数（A,B,C,D）用来计算信息摘要，初始化使用的是十六进制表示的数字。

A = 0x01234567

B = 0x89abcdef

C = 0xfedcba98

D = 0x76543210

（4）处理位操作函数。

X,Y,Z 为 32 位整数。

F(X,Y,Z) = X&Y|not（X）&Z

G(X,Y,Z) = X&Z|Ynot（Z）

H(X,Y,Z) = XxorYxorZ

I(X,Y,Z) = Yxor（X|not（Z））

（5）主要变换过程：使用常数组 T[1...64]，T[i]为 32 位整数用十六进制表示，数据用 16 个 32 位的整数数组 M[]表示。

具体过程如下：

```
/*处理数据原文*/
For I = 0 to N/15-1 do
/*每一次，把数据原文存放在 16 个元素的数组 X 中. */
For j = 0 to 15 do
Set X[j] to M[i*16+j].
end/结束对 J 的循环
/*Save A as AA, B as BB, C as CC, and D as DD. */
AA = A
BB = B
CC = C
DD = D
/*第 1 轮*/
/*以[a b c d k s i]表示如下操作
a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s).*/
/* Do the following 16operations.*/
[ABCD 0 7 1] [DABC 1 12 2] [CDAB 2 17 3] [BCDA 3 22 4]
[ABCD 4 7 5] [DABC 5 12 6] [CDAB 6 17 7] [BCDA 7 22 8]
[ABCD 8 7 9] [DABC 9 12 10] [CDAB 10 17 11] [BCDA 11 22 12]
[ABCD 12 7 13] [DABC 13 12 14] [CDAB 14 17 15] [BCDA 15 22 16]
/*第 2 轮*/
/*以[a b c d k s i]表示如下操作
a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s).*/
/* Do the following 16 operations. */
[ABCD 1 5 17] [DABC 6 9 18] [CDAB 11 14 19] [BCDA 0 20 20]
[ABCD 5 5 21] [DABC 10 9 22] [CDAB 15 14 23] [BCDA 4 20 24]
[ABCD 9 5 25] [DABC 14 9 26] [CDAB 3 14 27] [BCDA 8 20 28]
[ABCD 13 5 29] [DABC 2 9 30] [CDAB 7 14 31] [BCDA 12 20 32]
/* 第 3 轮*/
```

```
/* 以[a b c d k s i]表示如下操作
a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s). */
/* Do the following 16 operations. */
[ABCD 5 4 33] [DABC 8 11 34] [CDAB 11 16 35] [BCDA 14 23 36]
[ABCD 1 4 37] [DABC 4 11 38] [CDAB 7 16 39] [BCDA 10 23 40]
[ABCD 13 4 41][DABC 0 11 42] [CDAB 3 16 43] [BCDA 6 23 44]
[ABCD 9 4 45] [DABC 12 11 46] [CDAB 15 16 47] [BCDA 2 23 48]
/* 第 4 轮*/
/* 以[a b c d k s i]表示如下操作
a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s). */
/* Do the following 16 operations. */
[ABCD 0 6 49] [DABC 7 10 50] [CDAB 14 15 51] [BCDA 5 21 52]
[ABCD 12 6 53] [DABC 3 10 54] [CDAB 10 15 55] [BCDA 1 21 56]
[ABCD 8 6 57] [DABC 15 10 58] [CDAB 6 15 59] [BCDA 13 21 60]
[ABCD 4 6 61] [DABC 11 10 62] [CDAB 2 15 63] [BCDA 9 21 64]
/*然后进行如下操作*/
A = A + AA
B = B + BB
C = C + CC
D = D + DD
end/*结束对 I 的循环*/
```

（6）输出结果。

综上所述，MD5 算法是从 MD4 算法改进而来的，它在以下方面做了较大的改进。

- 增加了第四轮变换。
- 每一步都有唯一的加法常数。
- 为了减弱第二轮中函数 G 的对称性，从（（X^Y）V（X^Z）V（Y^Z））变为（（X^Z）V（Y^¬Z））。
- 每一步加上了上一步的结果，这将引起更快的雪崩效应。
- 改变了第二轮和第三轮中访问消息子分组的次序，使其形式更不相似。
- 近似优化了每一轮中的循环左移位移量以实现更快的雪崩效应。各轮的位移量互不相同。

4）SHA 算法

该算法首先将消息填充为 512 位的整数倍。填充方法与 MD5 完全一样：先添加一个 1，然后填充尽量多的 0 使其长度为 512 的倍数并余 448 位，最后 64 位表示消息填充前的长度。该算法有 5 个 32-位变量（与此相对应的是，MD5 算法仅有 4 个变量），这几个变量初始化为：

A = 0x67452301

B = 0xefcdab89

C = 0x98badcfe

D = 0x10325476

E = 0xc3d2e1f0

然后开始算法的主循环。它一次处理 512 位消息，循环的次数是消息中 512-位分组的数目。先把这 5 个变量复制到另外的临时变量当中：A 到 a，B 到 b，C 到 c，D 到 d，E 到 e。

主循环有四轮，每轮 20 次操作（MD5 有四轮循环，每轮 16 次操作）。每次操作对 abcde 中的三个进行一次非线性运算，然后进行与 MD5 中类似的移位运算和加运算。

该算法中，SHA 的非线性函数集如下（^表示与，V 表示或，¬表示反，⊕表示异或）：

Ft（X,Y,Z）=（X^Y）V（（¬X）^Z），其中，t=0～19

Ft（X,Y,Z）= X ⊕ Y ⊕ Z，其中，t=20～39

Ft（X,Y,Z）=（X^Y）V（X^Z）V（Y^Z），其中，t=40～59

Ft（X,Y,Z）= X ⊕ Y ⊕ Z，其中，t=60～79

该算法采用了以下四个常数。

Kt = 0x5a827999，其中，t=0～19

Kt = 0x6ed9eba1，其中，t=20～39

Kt = 0x8f1bbcdc，其中，t=40～59

Kt = 0xca62c1d6，其中，t=60～79

从以上对该算法的分析和描述可以看出，SHA 与 MD5 非常类似，但是它有 160-位散列值。主要的改变是添加了扩展转换，并且为了产生更快的雪崩效应而将上一轮的输出送至下一轮。从当前的情况来看，对 SHA 还没有已知的密码攻击，并且由于它产生 160-位散列，所以它比其他 128-位散列函数更能有效地抵抗穷举攻击（包括生日攻击）。然而，任何事情都是一分为二的，安全性得到提高，其所消耗的机器时间会随之增加，在下一节对这些算法的效率分析当中将会很好地说明这个问题。

### 3. 安装和配置 Tripwire

Tripwire 主要由策略和数据库组成。策略不仅指出 Tripwire 应检测的对象即文件和目录，而且还规定了用于鉴定违规行为的规则。一般情况下，对于/root、/bin 和/lib 目录及其中文件的任何修改都应视为违规行为。数据库则用来存放策略中规定的检测对象的快照。只要建立了策略和数据库，我们就可以随时用快照来比较当前的文件系统，然后生成一个完整性检测报告，从而判断系统的完整性是否受到攻击。除了策略和数据库外，Tripwire 还有一个配置文件，用以控制数据库、策略文件和 Tripwire 可执行程序的位置等。

为了防止被篡改，Tripwire 对其自身的一些重要文件进行了加密和签名处理。这里涉及两个密钥：site 密钥和 local 密钥。其中，前者用于保护策略文件和配置文件，如果多台机器具有相同的策略和配置的话，那么它们就可以使用相同的 site 密钥；后者用于保护数据库和报告，因此不同的机器必须使用不同的 local 密钥。

以下要给出该软件的安装以及使用步骤。要得到该软件可以从网站 http://sourceforge.net/project/tripwire/直接获得 tripwire-2.4.1.2-src.tar.bz2。

（1）解压缩安装文件到/usr/local 目录。

```
//切换工作路径
#cd /usr/local/

//解压缩
#tar xvfj tripwire-2.4.1.2-src.tar.bz2
```

（2）执行 make 命令，进行安装

```
//进入已经解压的文件夹
#cd tripwire-2.4.1.2-src

//生成 makefile 文件
#./configure

//执行 make 命令
#make

//执行 make install 命令
#make install
```

（3）生成基线数据库。

成功编译 Tripwire，就可以准备开始对需要监控的文件进行扫描，以生成 Tripwire 数据库，在 Tripwire 的 src 目录下，如下操作：

```
#./tripwire -init
```

（4）测试。

数据库生成了，使用命令运行 Tripwire，扫描系统的变化和细小改变。

```
#./tripwire—check
```

当第一次运行 Tripwire 时，需要进行一些准备工作，主要有编辑 config 文件、检查邮件报告是否正常、根据需要配置策略文件和初始化数据库文件，即创建一个签名的基线数据库。下一次运行时，它使用 twpol.txt 文件产生一个新的签名数据库。然后，比较两个数据库，实施用户定义的任何选项屏蔽（排除经常更改的文件），最后通过电子邮件或显示器来为用户在终端上输出一个可读的报告。

对于 Tripwire，为了保证其正确地运行，要特别注意其安装和配置的过程，具体步骤如下。

（1）创建密钥和签名。

在安装 Tripwire 之后，可以使用如下命令进行设置：

```
#./twinstall.sh
```

脚本 twinstall.sh 的作用在于执行下列任务。

- 创建 site 和 local 密钥，这时会要求输入口令（参见下面的步骤）。如果这两个密钥已存在，则可以跳过此步骤。其中，site 密钥存放在 site.key 文件中，而 local 密钥则存放在 hostname-local.key（这里的 hostname 是指该机器的主机名）文件之中。

- 利用 site 密钥对默认配置文件 twcfg.txt 进行签名，并将签名（而非被签名的文件 twcfg.txt）存放于文件 tw.cfg 之中。
- 利用 site 密钥对默认策略文件 twcfg.pol 进行签名，并将签名（而非被签名的文件 twcfg.pol）存放于文件 tw.pol 之中。

```
----------------------------------------------
The Tripwire site and local passphrases are used to
sign a variety of files, such as the configuration,
policy, and database files.

Passphrases should be at least 8 characters in length
and contain both letters and numbers.

See the Tripwire manual for more information.

----------------------------------------------
Creating key files...

(When selecting a passphrase, keep in mind that good passphrases typically
have upper and lower case letters, digits and punctuation marks, and are
at least 8 characters in length.)

Enter the site keyfile passphrase:
Verify the site keyfile passphrase:
Generating key (this may take several minutes)...Key generation complete.

(When selecting a passphrase, keep in mind that good passphrases typically
have upper and lower case letters, digits and punctuation marks, and are
at least 8 characters in length.)
Enter the local keyfile passphrase:
Verify the local keyfile passphrase:
Generating key (this may take several minutes)...Key generation complete.

----------------------------------------------
Generating Tripwire configuration file...

----------------------------------------------
```

```
Creating signed configuration file...
Please enter your site passphrase:
Wrote configuration file: /usr/local/etc/tw.cfg

A clear-text version of the Tripwire configuration file
/usr/local/etc/twcfg.txt
has been preserved for your inspection.  It is recommended
that you delete this file manually after you have examined it.



----------------------------------------------
Customizing default policy file...


----------------------------------------------
Creating signed policy file...
Please enter your site passphrase:
Wrote policy file: /usr/local/etc/tw.pol

A clear-text version of the Tripwire policy file
/usr/local/etc/twpol.txt
has been preserved for your inspection.  This implements
a minimal policy, intended only to test essential
Tripwire functionality.  You should edit the policy file
to describe your system, and then use twadmin to generate
a new signed copy of the Tripwire policy.
----------------------------------------------
The installation succeeded.
```

（2）编辑配置文件。

首先打开文本格式的配置文件 twcfg.txt。该文件的位置在前面所述安装过程后的/usr/local/etc 目录下。然后根据需要修改配置文件，修改完毕后存盘。最后使用 twadmin 命令根据已编辑的文本文件生成一个加密的配置文件。

```
ROOT               =/usr/local/sbin
POLFILE            =/usr/local/etc/tw.pol
DBFILE             =/usr/local/lib/tripwire/$(HOSTNAME).twd
REPORTFILE         =/usr/local/lib/tripwire/report/$(HOSTNAME)-$(DATE).twr
SITEKEYFILE        =/usr/local/etc/site.key
```

```
LOCALKEYFILE  =/usr/local/etc/localhost.localdomain-local.key
EDITOR       =/bin/vi
LATEPROMPTING =false
LOOSEDIRECTORYCHECKING =false
MAILNOVIOLATIONS =true
EMAILREPORTLEVEL =3
REPORTLEVEL   =3
MAILMETHOD    =SENDMAIL
SYSLOGREPORTING =false
MAILPROGRAM   =/usr/sbin/sendmail -oi -t



#twadmin --create-cfgfile --site-keyfile /etc/tripwire/site.key twcfg.txt
```

安装完毕后，该文件已存在，因此不必再重新创建。通常情况下，配置文件的内容不会发生变化，因此没有必要去修改它，使用 Tripwire 默认的就可以了。在此时我们应该测试一下 E-mail 报告功能是否起作用，以防以后遇到麻烦，输入以下命令进行测试。

```
#tripwire --test --mail user@domain.com
```

（3）编辑策略文件。

首先打开文本格式的策略文件 twpol.txt。该文件的位置在前面所述安装过程后的/usr/local/etc 目录下。Tripwire 在安装时已经在策略文件中写入了默认的检查规则。这些默认的规则主要检查重要的系统文件和 Tripwire 自身文件的完整性。

```
################################################ #
#                                       # #
#  Global Configuration Files (/etc/)         # #
#                                       ##
################################################
(
  rulename = "Global Configuration Files",
)
{
  /etc                         -> $(IgnoreNone) -SHa ;
  /etc/adjtime                 -> $(Dynamic) ;
  /etc/aliases.db              -> $(Dynamic) ;
  /etc/bashrc                  -> $(Dynamic) ;
  /etc/csh.cshrc               -> $(Dynamic) ;
  /etc/csh.login               -> $(Dynamic) ;
```

```
    /etc/mail/statistics          -> $(Growing) ;
    /etc/profile                  -> $(Dynamic) -i ;
    /etc/mtab                     -> $(Dynamic) -i ;
    /etc/rc.d                     -> $(IgnoreNone) -SHa ;
    /etc/sysconfig                -> $(IgnoreNone) -SHa ;
    /etc/sysconfig/hwconf         -> $(Dynamic) -m ;
}
```

由于默认的配置不能监视系统中的 SUID 和 SGID 文件，而这对于我们的系统安全是非常重要的，因此，我们需要修改配置，加入对该项目的监视。使用如下命令可以找出系统中的所有 SUID 文件：

```
#find / -type f -perm -4000 -print
```

以下命令可以找出系统中的所有 SGID 文件：

```
#find / -type f -perm -2000 -print
```

现在，我们需要把以上命令找出的 SUID 和 SGID 文件加入到 Tripwire 的策略文件中去。除此之外，我们还要把所有用户 home 目录下的初始文件也纳入监视的范畴。主要包括以下文件：

- .bashrc、.profile、.bash_profile、.cshrc、kshrc、.login 等。
- bash、csh 以及 ksh 登录之后的初始化命令或者脚本。
- .forward：告诉/usr/lib/sendmail 把邮件转发到某些地址。
- .rhosts：可以使用 rsh 登录到本账户的远程主机名。
- .xinitrc、.Xauthority、Xdefault 等 X 窗口系统的一些重要文件。

在创建 Tripwire 的特征码数据库之前，还有一件事情要做，就是检查.netrc 和.rhosts 文件的权限是否为 600。修改完策略文件后存盘。再使用 twadmin 命令根据已编辑的文本文件生成一个加密的策略文件。最后，策略文件的文本文件要删除掉，否则该文件的内容易被查看。

```
#twadmin --create-polfile twpol.txt
```

安装完毕后，该文件已存在，因此不必再重新创建。

（4）生成基线数据库。

配置文件和策略文件都编辑和生成好了之后，就应该根据配置文件的规则生成基线数据库。基线数据库在 Tripwire 安装完毕后生成一次即可。我们使用 Tripwire 命令来生成基线数据库。

```
#tripwire --init
```

基线数据库生成时，Tripwire 会提示输入 local key，对其进行高强度的加密，以防止对文件内容的非法改变。

（5）运行完整性检查。

基线数据库生成之后，我们可以使用 tripwire 命令随时进行完整性检查。

```
#tripwire --check
```

进行检查时可以指定检查报告的存储位置。其命令为：

```
#tripwire --check --twfile/var/lib/report/report.twr
```

进行检查时也可发送 E-mail 报告结果。其命令为：

```
#tripwire --check --email-report
```

进行检查时指定使用 E-mail 进行发送的报告等级。其命令为：

```
#tripwire --check --email-report --email-report-level 2
```

使用指定严重性等级的规则进行检查。其命令为：

```
#tripwire --check --severity 80
```

使用指定的规则名进行检查。其命令为：

```
#tripwire --check --rule-name rulename
```

只检查指定的文件或目录。其命令为：

```
#tripwire --check object1 object2 object3 …
```

进行检查时忽略某些属性（因为有些属性的检查比较耗费系统资源，比如 Hash 算法）。其命令为：

```
#tripwire --check --ignore "property, property, property, property"
```

如果完整性检查完毕后，发现 E-mail 报告功能未生效，可以检查两个方面：一个是策略文件中规则的 emailto 属性必须填写妥当，另一个是运行 Tripwire 命令时，--email-report 选项必须被包含。

（6）查阅报告。

完整性检查进行完毕后，我们就可以查阅报告以发现有哪些文件遭到了改动，改动了什么。使用 twprint 命令可以输出报告，它有多种使用方式。

如下命令将加密的报告内容输出到显示器：

```
#twprint --print-report --twrfile /var/lib/report/report.twr
```

如下命令将加密的报告内容输出到一个文本文件：

```
#twprint --print-report --twrfile /var/lib/report/report.twr - >myreport.txt
```

如下命令输出报告时指定输出的报告等级：

```
#twprint --print-report --report-level 4 --twrfile /var/lib/report/
report.twr
```

（7）升级基线数据库文件。

如果在报告中发现了一些违反策略的错误，而这些错误又被认为是正常的，那就要使用 Tripwire 命令更新基线数据库：

```
#tripwire --update --twrfile /var/lib/report/report.twr
```

也可以在进行完整性检查之后立即自动进行更新，其命令如下。

```
#tripwire --check --interactive
```

（8）升级策略文件。

随着系统的变化，原来的策略文件将不能满足需要，因此必须要不断地更新策略文件中的规则。更新和创建新的策略文件不同，因为如果为 Tripwire 创建了新的策略文件，那么就必须要重新生成基线数据库。更新时首先打开策略文件的文本文件：

```
#twadmin --print-profile > twpol.txt
```

然后编辑该文件，完毕后存盘。最后使用 Tripwire 命令进行策略更新。

```
#tripwire --update-policy twpol.txt
```

在此步骤中，Tripwire 软件会要求输入 site key。

（9）改变 site key 和 local key。

site key 和 local key 是在安装时生成的，但是也可以随时修改。注意，如果已经用来加密的密钥文件被删除了或是被覆盖了，那么 Tripwire 加密过的文件都不能访问了。因此，我们要对这两个文件做备份。很多时候会发现我们的口令可能不太安全，因此要改变口令。需执行以下命令即可：

```
#twadmin --generate-keys --local-keyfile /etc/tripwire/site.key
#twadmin --generate-keys --local-keyfile /etc/tripwire/local.key
```

但是这么做的话，会造成使用以前密钥进行加密的文件无法打开的情况。我们要使用以前的策略文件、配置文件、数据库文件、报告文件的话，因此在改变口令之前，我们必须使用以下的命令把这些已加密的文件进行解密。

```
#twadmin --remove-encryption file1 file2 ...
```

在生成新的密钥文件之后，我们还应该用新密钥对这些文件进行加密。配置文件和策略文件只能用 site key 加密，而数据库文件和报告文件只能用 local key 加密。

```
#twadmin --encrypt --site-keyfile /etc/tripwire/site.key file1 file2 file3 …
#twadmin --encrypt --local-keyfile /etc/tripwire/local.key file1 file2 file3 …
```

### 4. 使用 Tripwire 扫描文件系统改变

下面就是着手来对文件系统的改变或者是不经意的"破坏"甚至是黑客的"篡改"来进行扫描和发现了。

（1）设定监控目标文件/目录。

虽然 Tripwire 是一个功能和性能非常优秀的开源软件，但是从前面的介绍不难看出：Tripwire 主要依赖的是哈希算法来比较前后两次文件的哈希值，如果有所改变则表明文件被改动了。况且，Tripwire 的策略文件 twpol.txt 默认的监控范围是整个文件系统，包含成千上万个目录，而目录里面包含了成百万的文件，因此在一般的机器上需要花费很长的时间来进行扫描和形成基线数据库。为了采用 Tripwire 来实时或者快速地洞悉文件系统的"蛛丝马迹"变动，则需要使用者对监控目标进行高度地挑选和精简。其实一言以概之，就是合理地选择部分需要监控的目标。

因此我们此处对其简化，仅仅扫描/home/liyang 目录，如下所示的策略文件：

```
##############################################
 #                                    ##
############################################## #
#                                    # #
#  Monitor Filesystems                  # #
#                                    ##
##############################################
(
  rulename = "Monitor Filesystems",
)
```

```
{
  /home                        -> $(ReadOnly) ;  # Modify as needed
}
```

然后使用如下命令对其进行更新：

```
twadmin --create-polfile twpol.txt
```

（2）生成初始基线数据库（Baseline）。

使用 tripwire-init 命令生成初始基线数据库，如图 5-3 所示。

（3）实施扫描，查看报告。

我们对/home 目录下的文件和目录进行了修改，生成了两个新的文件，使用如下命令得到检查结果：

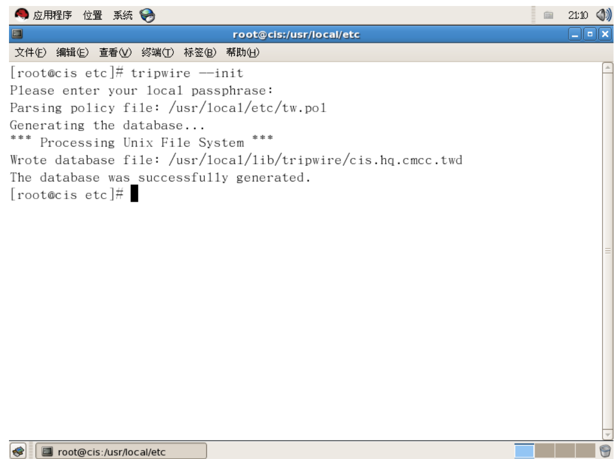



图 5-3　生成基线数据库

```
# tripwire -check

Parsing policy file: /usr/local/etc/tw.pol
*** Processing Unix File System ***
Performing integrity check...
Wrote report file: /usr/local/lib/tripwire/report/cis.hq.cmcc-20100220
-211310.twr


Open Source Tripwire(R) 2.4.1 Integrity Check Report

Report generated by:          root
Report created on:            2010年02月20日 星期六 21时13分10秒
Database last updated on:      Never

=========================================================================
Report Summary:
=========================================================================
```

```
Host name:                 cis.hq
Host IP address:            10.1.11.143
Host ID:                   None
Policy file used:           /usr/local/etc/tw.pol
Configuration file used:     /usr/local/etc/tw.cfg
Database file used:          /usr/local/lib/tripwire/cis.hq.cmcc.twd
Command line used:           tripwire --check

===========================================================================
Rule Summary:
===========================================================================

---------------------------------------------------------------------------
  Section: Unix File System
---------------------------------------------------------------------------

  Rule Name                    Severity Level    Added    Removed  Modified
  ---------                    --------------    -----    -------  --------
* Monitor Filesystems                0              3        0         1
  (/home)

Total objects scanned:  13
Total violations found:  4

===========================================================================
Object Summary:
===========================================================================

---------------------------------------------------------------------------
# Section: Unix File System
---------------------------------------------------------------------------

---------------------------------------------------------------------------
Rule Name: Monitor Filesystems (/home)
Severity Level: 0
```

```
--------------------------------------------------------------------------


Added:
"/home/tripwire_test"
"/home/tripwire_test1"
"/home/tripwire_test2"

Modified:
"/home"

==========================================================================
Error Report:
==========================================================================


No Errors


--------------------------------------------------------------------------
*** End of report ***

Open Source Tripwire 2.4 Portions copyright 2000 Tripwire, Inc. Tripwire is
a registeredtrademark of Tripwire, Inc. This software comes with ABSOLUTELY NO
WARRANTY;for details use --version. This is free software which may be
redistributedor modified only under certain conditions; see COPYING for details.
All rights reserved.
Integrity check complete.
```

通过上面的结果可以看出，Tripwire 把我们对目录的修改全部记录下来了，并且还记录了系统自身文件发生变化的某些细节。最重要的一点是：Tripwire 已经将这些细小的变化都记录了下来，使得诸如 Web 服务、电子商务等网络管理员能够有的放矢地进行维护工作，而不是“胡子眉毛一把抓”，从而可以高效、方便地进行网络管理，适时地对重要网络系统的文件进行恢复，这也正是 Tripwire 在网络安全日益热门的今天，能够高频率地出现在服务器应用中的最重要原因。

鉴于 Tripwire 是一种非常优秀的保证文件系统安全的工具，在使用中有许多经验、技巧可以借鉴，特总结如下供读者使用。

- 把握好运行和安装 Tripwire 的时机。最好的时机是在刚刚建立好系统之后，不论是新建的还是升级的。一个运行了一段时间的系统可能已经被破坏了，之后运行 Tripwire 将几乎没有任何价值。如果不可能在系统刚安装或更新的时候安装 Tripwire，那么考虑以单用户模式重新安装系统文件，这是保证系统纯净的惟一方法。
- 切实保证基线数据库的安全。用户需要把基线数据库同程序文件一起保存在一个安全

的地方，以免攻击者更改文件而使该程序达不到监控的目的。最优的安排是把这些文件保存在一个移动的只读介质中，比如写保护的软盘、光盘或潜在攻击者不能访问的 CD-ROM；还可以把文件存放在一个安全的远程及其分区中并以只读方式输出。记住采用 mount 命令使一个硬盘只读是不够的，因为一个有 root 权限的攻击者可以轻易地把硬盘改为可写。

- 明确和缩减需要监控的文件对象。将需要监控的关键和核心文件进行整理和分类，以方便 Tripwire 的实时监控。并且，由于默认的 Tripwire 的监控文件数量巨大（具体可以参见/usr/local/etc/twcfg.txt 文件），如果按照默认的文件及目录进行扫描和监控，非常消耗 CPU 的时间以及用户时间。所以，建议读者在实际使用过程中修改和缩减/usr/local/etc/twcfg.txt 中的 Tripwire 扫描及监控范围，从而高效地发挥 Tripwire 的功能。

### 5.3.7 根用户安全管理

通常用户以 root 用户名登录，或通过给出 su 命令并提供 root 密码来获得 root 权限。最近 sudo 的使用已经接管了这个获得 root 权限的经典技术。使用 sudo，用户以自己的用户名登录后，给出一个 sudo 命令，并提供自己的密码（非 root 密码），以获得 root 权限。

当一个普通用户在图形化环境中执行一个特权命令，系统会提示输入 root 密码或用户密码，这取决于系统是如何设置的。有些发行版通过不分配 root 口令来锁定 root 账户。在这些系统中，不能通过提供 root 密码获得 root 权限。Fedora/RHEL 在安装系统时会分配 root 密码，以便从一开始就可以使用这些技术。

安装时，某些系统（非 Fedora/RHEL）通过不提供 root 口令来锁定 root 账户。此设置可以防止任何人以 root 账户登录（除非当把系统带到单用户模式时）。然而，/etc/passwd 文件的第一行的 root 用户名表明有 root 账户。此账户/用户拥有文件（发出 ls-l /bin 命令）并运行进程（发出 ps-ef 命令，看看左边输出的列）。该 root 账户是 Linux 系统运转的关键。

正确设置后，sudo 实用程序便可以像 root 用户一样运行命令。本文"使用 root 特权"短语强调的是，虽然不以 root 身份登录，但当使用 sudo 时，就拥有了 root 用户的特殊权力。

以下描述了一些可以获得或授予 root 特权的方法。其中一些技术需要提供 root 账户密码。同样，如果 root 账户被锁住，就不能使用这些技术，除非解开了 root 账户（设置 root 密码）。其他技术依靠设置 sudoers 文件，以获得 root 特权。如果没有按这种方式设置此文件，就不能使用这些技术，除非设置 sudoers 文件。

- 当把系统带到单用户/救援模式时，登录的用户名就是 root。
- 当以自己的用户名登录时，可以使用 su（substitute user：替代用户）命令。然后提供 root 密码，将以 root 权限运行。
- sudo 实用程序允许指定的用户以 root 特权运行选定的命令，而他们都以其自身登录。可以设置 sudo 以允许特定用户执行需要 root 特权的具体任务，而无须授予这些用户对整个系统的 root 特权。
- 一旦系统启动并运行在多用户模式中，可以登录为 root 用户。然后提供 root 密码，将以 root 特权运行。
- 某些程序在启动时需要密码（其密码或 root 密码，由命令和系统配置而定）。当提供密码，程序以 root 特权运行。当退出该程序时，停止以特权用户运行。当不需要或不

打算一直以 root 特权登录，此设置可防止这样做。

- 任何用户都可以创建一个 setuid（设置用户 ID）文件。setuid 程序代表文件的所有者运行，拥有所有者所具备的全部访问特权。当使用 root 特权，可以将 root 所拥有的文件的权限更改到 setuid。当一个普通用户执行一个由 root 拥有的文件，并且具有 setuid 权限，则该程序具有有效的 root 特权。换句话说，该程序可以做 root 特权能做而一般程序不能做的任何事情。用户的特权不会改变。因此，当程序运行完毕后，所有用户的特权仍和程序启动之前一样。由 root 所拥有的 setuid 程序都非常强大并对系统安全极其危险，这就是为什么系统很少包含这些程序的原因。root 用户所拥有的 setuid 程序的例子包括 passwd、at 和 crontab。

由于 root 拥有的 setuid 程序允许不知道 root 密码的人，不用 sudo 就可以取得 root 特权，它们是恶意用户的诱人目标。此外，使正常程序崩溃的编程错误可以在 setuid 程序中变成 root 漏洞。系统应尽可能少有这些程序。通过使用 nosuid 选项挂载一个文件系统，可以在文件系统级别禁用 setuid 程序。还可以使用 SELinux 禁用 setuid 程序。在 Fedora/RHEL 未来的版本中将删除所有的 setuid 文件，参阅 fedoraproject.org/wiki/Features/RemoveSETUID。

值得提醒的是，不要在互联网上允许 root 访问。禁止用户通过网络以 root 登录是 Fedora/RHEL 的默认策略。/etc/securetty 文件必须包含希望用户能够以 root 身份登录的所有设备名称。然而，可以使用 ssh 通过网络以 root 登录，因为 Fedora/RHEL 中安装的 ssh 不遵循 securetty 或 access.conf 中的指示。此外，在/etc/ssh/sshd_config 中，Fedora/RHEL 将 PermitRootLogin 设置为 yes（默认设置），以允许 root 使用 ssh 登录。为了使系统更安全，将 PermitRootLogin 更改为 no。

#### 1. 使用 su 获取 root 特权

安装 Fedora/RHEL 时，用户为 root 账号分配一个密码。因此，可以使用 su 获得 root 特权，而无须做进一步的设置。

su 工具可以产生一个 shell，或者执行一个指定用户（包括 root）身份和特权的程序。

- 在命令行上跟随 su 之后的是用户名；如果使用 root 特权或知道用户的密码，然后将具有该用户的身份。
- 当发出一个不带参数的 su 命令时，su 默认产生一个具有 root 特权（必须提供 root 密码）的 shell。

当发出 su 命令以使用 root 特权，su 产生一个新的 shell，显示#提示符。可以终止这个 shell：按 Control-D 或发出 exit 命令返回到正常状态（之前的 shell 和提示符）。不带任何参数的 su 命令会改变用户和组 ID，但对环境的改变却是最小的。例如，发出 su 命令之后，PATH 值没有变化。ID 实用程序显示用户和组 ID 以及关联的组。在下面的例子中，从 context 开始的信息涉及 SELinux：

```
$ pwd
/home/sam
$ echo $PATH
/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/home/sam/bin
$ id
uid=500(sam) gid=500(sam) groups=500(sam) context=unconfined_u: ...
```

```
$ su
Password:
# pwd
/home/sam
# echo $PATH
/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/home/sam/bin
# id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm) ...
# exit
Exit
$
```

当给出 su-命令（可以用-l 或-login 替换连字符），su 提供一个 root 登录 shell：就好像以 root 身份登录一样。不仅 shell 的用户和组 ID 匹配 root，而且环境也与以 root 身份登录时相同。登录 shell 在显示提示符之前执行相应的启动文件，并且工作目录也被设置为同以 root（/root）身份登录时的工作目录。PATH 也被设置为好像以 root 身份登录一样，通常在/bin 和/usr/bin 之前包含/sbin 和/usr/sbin 目录。

```
$ su -
Password:
# pwd
/root
# echo $PATH
/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin
```

可以使用带-c 选项的 su 以 root 特权运行一个命令，当命令执行完毕后返回到原来的 shell。在下面的例子中，Sam 尝试以自己的身份（非特权用户）设置系统时钟。在 Sam 输入 date 之后，date 实用程序显示错误消息。下面是 Sam 所输入日期的扩展版。当他使用 su 运行 date 以设置系统时钟时，su 会提示他输入密码，他输入 root 密码，然后命令成功。引号是必要的，因为 su-c 把该命令作为单个参数执行。

```
$ date 12281448
date: cannot set date: Operation not permitted
Tue Dec 28 14:48:00 PST 2010
$ su -c "date 12281448"
Password:
Tue Dec 28 14:48:00 PST 2010
```

下面的例子首先表明，不允许 Sam 终止过程。然而，随着 su-c 和 root 密码的使用，Sam 使用 root 特权就可以终止进程。

```
$ kill -15 4982
-bash: kill: (4982) - Operation not permitted
$ su -c "kill -15 4982"
Password:
```

```
$
```

最后一个例子将和-c 选项结合，以显示如何在 root 环境中使用 root 特权运行一个命令：

```
$ su -c pwd
Password:
/home/sam
$ su - -c pwd
Password:
/root
```

使用 root 特权时，保存在 PATH 中的目录越少，就越有可能使用 root 特权执行一个不受信任的程序。如果可能的话，在 root 的 PATH 中只保留默认目录，如/sbin 和/usr/sbin。一定不要把工作目录包含在 PATH 中（如：.或: :在 PATH 中的任何地方，或作为 PATH 中最后一个元素）。

#### 2. 使用 sudo 获取 root 特权

如果 sudo（www.sudo.ws）被设置为不能使用，但知道现有 root 密码，参考下面的“快速设置”说明：关于设置 sudo 以便可以使用它获得 root 特权。

使用 sudo，而非 root 账户，进行系统管理，具有许多优点。

- 当运行 sudo，需要的密码并非 root 密码，这样只需记住一个密码即可。
- sudo 实用程序记录其执行的所有命令。如果犯了一个错误，此日志可用于系统审计来追溯的操作。
- sudo 实用程序记录发出 sudo 命令的用户。在有多个管理员的系统上，该日志告诉哪些用户发出了 sudo 命令。没有 sudo，就无从知道哪个用户使用 root 特权发出一个命令。
- sudo 实用程序比使用 su 和 root 账户可以实现更细粒度的安全策略。使用 sudo，可以使特定用户执行特定命令，而使用经典的 root 账户设置，就不能这样做。
- 使用 sudo 使得恶意用户更难获得对系统的访问权。当有一个未锁定的 root 账户时，恶意用户一开始就知道他想破解的账户名。当 root 账户被锁住时，该用户为进入系统就不得不先确定用户名和密码。
- 管理许多系统的 root 密码是具有挑战性的。如果不写下来（并把它们存储在安全的地方），记住每个系统的密码是很难的，并且检索密码也很耗时。使用 sudo，即使是完全的 root shell 访问，使得在大量的系统上获得 root 特权，并跟踪每个系统上获得 root 特权的用户，使这样的任务变得更容易。

有些用户质疑 sudo 不如 su 安全。因为都依靠密码，它们共享相同的长处和短处。如果密码被攻破，系统就会大打折扣。但是，如果 sudo 允许执行一项任务的用户密码被破解，系统整体不会有风险。因此，如果使用得当，sudo 权限结构的更细粒度使其比 su 更安全。

对于设置系统来说，使用 sudo 可能并不总是最好的、安全的方式。在由单个用户使用的系统上，使用 sudo 与小心使用 su 和 root 密码之间没有太大的区别。相比之下，在有多个用户的系统上，特别是在具备中央管理的系统网络上，sudo 可以设置得比 su 更安全。

以下行在/etc/sudoers 文件中，允许 wheel 组的成员使用 sudo 来取得 root 特权：

```
%wheel ALL=(ALL) ALL
```

如果注释掉此行（在 RHEL 和 Fedora15 之前的版本中），采用 root 特权，使用 visudo 来删

除前导哈希标记（#），以便 wheel 组的成员能获得 root 特权。

下一步就是使用 root 特权，运行 usermod-a 和 usermod-G wheel 命令，把被授予 root 特权的用户名添加到 wheel 组中。在下面的例子中用 sam 替代该用户名。

```
# usermod -a -G wheel sam
# grep wheel /etc/group
wheel:x:10:root,sam
```

默认情况下，首次运行时，sudo 会要求输入密码（非 root 密码）。此时，sudo 设置了时间戳。提供密码后，基于该时间戳，五分钟内 sudo 不会再次提示输入密码。

在下面的例子中，Sam 尝试以自己的身份（非特权用户）设置系统时钟。date 实用程序显示错误消息，后边是 Sam 所输入日期的扩展版。当他使用 sudo 运行 date 以设置系统时钟时，sudo 提示他输入其密码，然后命令成功。

```
$ date 01121500
date: cannot set date: Operation not permitted
Wed Jan 12 15:00:00 PST 2011
$ sudo date 01121500
[sudo] password for sam:
Wed Jan 12 15:00:00 PST 2011
```

接下来，Sam 使用 sudo 卸载文件系统。因为他是在之前 sudo 命令的五分钟内发出该命令，因此无须提供密码：

```
$ sudo umount /music
$
```

现在，Sam 使用-l 选项来检查 sudo 允许运行哪个命令。由于按照前面"快速设置"的解释来设置 sudoers 文件，他被允许以任何用户运行任何命令。

```
$ sudo -l
...
```

用户sam可能在此主机上运行以下命令：

```
(ALL) ALL
```

用户可以授予 root 特权来编辑一个文件，主要包括以下情况。

- 使用-e 选项，或被用作 sudoedit 时，sudo 以 root 特权编辑其参数命名的文件。默认情况下，sudo 使用 vi 编辑器。
- 任何以 root 特权运行命令的用户都可以使用-e 选项。要给其他用户使用 root 特权编辑任何文件的权限，需要在 sudoers 文件中指定该用户可以执行 sudoedit 命令。
- 以这种方式调用编辑器，在该用户环境以最少特权运行此编辑器。sudo 实用程序第一次会把要编辑的文件复制为属于该用户的一个临时文件。如果该文件不存在，sudo 创建一个属于该用户的新文件。一旦用户编辑完该文件，sudo 将其复制回原来的地方（并恢复为原来的权限）。

当有几个需要以root特权运行的命令，生成一个root shell可能会更容易，只需发出该命令而不必在每个命令前面键入sudo，并从shell退出。这种技术会破坏一些内置入sudo的保护，所

以要慎重使用，记得尽快返回到非root shell。使用sudo-i生成root shell：

```
$ pwd
/home/sam
$ sudo -i
# id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys) ...
# pwd
/root
# exit
Logout
$
```

在这个例子中，sudo派生了一个root shell，它显示一个#提示符提醒Sam他正在以root特权运行。id实用程序显示该用户运行shell的身份。exit命令（也可以使用Control-D组合）终止root shell，将Sam返回到其正常状态和他之前的shell和提示符。

在前面的例子中，pwd内建命令显示由-i选项创建的环境修改后的样子。此选项派生一个root登录shell（与以root身份登录的用户环境相同的shell），并执行root的启动文件。在发出sudo -i命令之前，pwd内建命令显示/home/sam为Sam的工作目录；执行该命令后，它显示/root（root的主目录）为其工作目录。使用-s选项派生一个root shell，而无须修改环境。当调用不带选项的sudo时，它运行在未修改的环境中指定的命令。为了展示此功能，下面的例子调用不带选项的sudo来运行pwd。以这种方式运行，命令的工作目录不会发生改变。

```
$ pwd
/home/sam
$ sudo pwd
/home/sam
```

因为虽然该sudo派生的shell以root特权执行ls，该用户正在运行的非特权shell将输出进行了重定向，用户的shell无权写入到/root，所以下面的命令将会失败。

```
$ sudo ls > /root/ls.sam
-bash: /root/ls.sam: Permission denied
```

有几种方法解决这个问题。最简单的是将整个命令行传递到sudo中运行的shell：

```
$ sudo bash -c "ls > /root/ls.sam"
```

bash-c选项派生了一个shell，来执行该选项后边的字符串，然后终止。sudo实用程序以root特权运行这个shell。必须引用该字符串，以防止非特权shell解释其中的特殊字符。还可以使用sudo-i生成一个root shell来执行该命令，然后从特权shell退出。

可以使用命令行选项来控制sudo如何运行命令。以下是sudo命令行的语法：

```
sudo [options] [command]
```

其中options是一个或多个选项，command是要执行的命令。如果没有-u选项，sudo以root特权运行command。

下面是一些较常见的options（选项）。完整列表请参阅sudo的手册页。

**- b**

（background）在后台运行command。

**- e**

（edit），使用此选项，command是一个文件名，而不是命令。此选项会使sudo以root特权，使用由SUDO_EDITOR、VISUAL或EDITOR环境变量命名的编辑器，来编辑command文件。默认是vi编辑器。另外，还可以使用不带任何选项的sudoedit实用工具。使用这种技术，编辑器不会以root特权运行。

**- i**

（initial login environment）为root派生在/etc/passwd中（或为由-u指定的另一个用户）指定的shell，运行root（或其他用户的）的启动文件，一些例外（例如，不改变TERM）。不要带command。

**- k**

（kill）重置该用户运行此命令的时间戳，这意味着该用户下次运行sudo时必须输入其密码。

**- L**

（list defaults）列出在sudoers文件中预设行设置的参数。不要带command。

**- l**

（list commands）列出允许运行sudo的用户在本地系统上运行的命令。不要带command。

**–s**

（shell）产生一个如在/etc/passwd文件中指定的新root（或为由-u指定的另一个用户）shell。与-i类似，但不改变环境。不要带command。

**- u user**

以user特权运行command。没有这个选项，sudo以root特权运行command。

### 3. 配置 sudo

安装后，只有仔细配置，sudo才会安全和可靠。sudo配置文件是/etc/sudoers。可以编辑该文件，以赋予特定用户只能够以root特权运行某些特定命令（而非任何命令）的能力。还可以限制这些用户只能使用某些特定的选项或参数运行命令。或者可以设置sudo，使一个特定用户当以root特权运行命令时，不能使用一个特定的选项或参数。

编辑sudoers的最佳方法是在命令中这样使用visudo：su -c visudo命令或sudo visudo。visudo实用程序锁定、编辑和检查sudoers文件的语法。默认情况下，visudo调用vi编辑器。可以设置SUDO_EDITOR、VISUAL或EDITOR环境变量，以使visudo调用不同的编辑器。以下命令使visudo调用nano编辑器（nano包）：

```
$ export EDITOR=$(which nano)
```

用所选择的文本型编辑器替换nano。把这个命令放在启动文件中，以每次登录时自动设置此变量。值得注意的是，需要始终使用visudo编辑sudoers文件。

sudoers文件中的一个语法错误，就会使得无法使用sudo来取得root特权。如果直接编辑这个文件（不使用visudo命令），直到发现不能使用sudo，才知道引入了语法错误。visudo实用程序检查sudoers的语法后，才允许退出。如果它找到了一个错误，会给修复错误的选择，不保存对文件的更改并退出；或保存对文件的更改并退出。最后一个选择通常是较差的，所以visudo用（DANGER!）标记它。

在sudoers文件中，注释以#号开头，可以出现在一行的任何地方。除了注释，这个文件包含三种类型的条目：用户特权规范、别名和默认值。这些条目各占一行。可以通过使用反斜杠（\）来继续一行。

指定用户特权的行格式如下（等号两边的空白是可选的）：

```
user_list host_list = [(runas_list)] command_list
```

- user_list 指定本规范行适用的用户。该列表可以包含用户名、组（%前缀）以及用户别名（下一节）。可以使用内置命令 alias ALL 来使该行适用于所有用户。
- host_list 指定本规范行适用的主机。该列表可以包含一个或多个主机名、IP 地址或主机别名。可以使用内置命令 alias ALL 来使该行适用于所有系统，参考 sudoers 文件。
- runas_list 指定运行 command_list 中命令的用户，当带-u 选项调用 sudo 时。该列表可以包含用户名、组（%前缀）以及使用 runas 别名（在下一节讨论）。它必须括在括号内。没有 runas_list，sudo 假定为 root。可以使用内置命令 alias ALL 来使该行适用于所有的用户和组。
- command_list 指定本规范行适用的实用程序。此逗号分隔的列表可以包含实用程序名、实用程序的目录名以及命令别名（在下一节讨论）。所有名称必须是绝对路径名；目录名必须以一个斜杠（/）结束。如果命令前面加上一个惊叹号（!），则是排除该命令。可以使用内置命令 alias ALL 来使该行适用于所有的命令。

在command_list中包含的字符串sudoedit，赋予user_list中的用户使用root特权编辑文件的权限。如果对command_list中的命令用个双引号（" "）引起来，用户将无法指定任何命令行参数，其中包括该命令的选项。或者可以指定参数，包括通配符，来限制允许用户对该命令使用的参数。

以下用户特权规范允许Sam在所有系统（由ALL指定）上使用sudo来挂载和卸载文件系统（以root特权运行mount和umount），参考包含本规范的sudoers文件：

```
sam ALL=(root) /bin/mount, /bin/umount
```

（root）runas_list是可选的。如果省略它，sudo允许用户以root特权运行command_list中的命令。在下面的例子中，Sam要利用这些特权。他不能直接运行umount，而是必须调用sudo来运行它。

```
$ whoami
Sam
$ umount /music
umount: only root can unmount /dev/sdb7 from /music
$ sudo umount /music
[sudo] password for sam:
$
```

如果用下面的一行替换上述sudoers中的该行，Sam就无法卸载/p03，尽管他仍然可以卸载其他任何文件系统，并可以安装任何文件系统：

```
sam ALL=(root) /bin/mount, /bin/umount, !/bin/umount /p03
```

sudoers中前面行的结果随后显示。sudo程序不提示输入密码，因为Sam在最后五分钟内输过自己的密码。

```
$ sudo umount /p03
Sorry, user sam is not allowed to execute '/bin/umount /p03' as root on
```

```
localhost.
```

以下行限制Sam挂载和卸载已挂载在/p01、/p02、/p03 和/p04 上的文件系统：

```
sam ALL= /bin/mount /p0[1-4], /bin/umount /p0[1-4]
```

以下命令显示的结果是：

```
$ sudo umount /music
Sorry, user sam is not allowed to execute '/bin/umount /music' as root on
localhost.
$ sudo umount /p03
$
```

在sudoers中的以下行允许wheel组成员的用户使用sudo来取得root特权：

```
## Allows people in group wheel to run all commands
%wheel ALL=(ALL) ALL

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL
```

此用户特权规范适用于所有系统（由等号左边的ALL表示）。如注释所表明的，此行允许wheel组的成员（组名前面加一个百分号指定：%wheel）以任何用户（括号内的ALL）运行任何命令（最右边的ALL）。当不带–u选项调用sudo实用程序，sudo以root特权运行指定的命令，大部分时间是这样使用sudo的。

如果将前面sudoers中的行按如下修改，它会允许wheel组的成员以任何用户运行任何命令，但有一个例外：不允许他们运行passwd来更改root密码（虽然他们可以获得root特权并能手动编辑它）。

```
%wheel ALL=(ALL) ALL, !/usr/bin/passwd root
```

别名可让重命名和/或对用户、主机或命令进行分组。以下是别名定义的格式：

```
alias_type alias_name = alias_list
```

其中alias_type是别名的类型（包括User_Alias、Runas_Alias、Host_Alias、Cmnd_Alias），alias_name是别名的名称（约定为全部大写字母），alias_list是一个逗号分隔的构成别名的一个或多个元素的列表。别名元素前面的感叹号（!）是对它的否定。

**User_Alias**

用户别名的alias_list与用户特权规范（在上一节讨论）的user_list相同。来自sudoers文件的下面的几行定义了三个用户别名：OFFICE、ADMIN和ADMIN2。第一行别名定义的alias_list包括用户名zach、sam和sls；第二行包括两个用户名和admin组的成员；第三行包括除Max外的admin组的所有成员。

```
User_Alias OFFICE = zach, sam, sls
User_Alias ADMIN = max, zach, %admin
User_Alias ADMIN2 = %admin, !max
```

**Runas_Alias**

runas别名的alias_list与用户特权规范（在上一节讨论）的runas_list相同。下面的SM runas别名包括sam和sls用户名：

```
Runas_Alias SM = sam, sls
```

**Host_Alias**

主机别名仅在sudoers文件被运行多个系统的sudo引用时才有意义。主机别名的alias_list与用户特权规范（在上一节讨论）的host_list相同。以下行定义的LCL别名包括名为guava和plum的系统：

```
Host_Alias LCL = guava, plum
```

如果想在此列表中使用完全合格的主机名（hosta.example.com而不只是hosta），必须设置fqdn标志。但是，这样做可能会降低sudo的性能。

**Cmnd_Alias**

命令别名的alias_list与用户特权规范的command_list相同。下面的命令别名包括三个文件，包含在尾部以/表示的目录中，该目录包括所有这样的文件：

```
Cmnd_Alias BASIC = /bin/cat, /usr/bin/vi, /bin/df, /usr/local/safe/
```

**Defaults (Options)**

通过使用Defaults关键字，可以更改配置选项的默认值。在此列表中的大部分值是隐式布尔型（on或off）或字符串型的标志。在一个默认行上命名一个标志以打开它，在其前面加上一个!以关闭它。下面的sudoers文件中的一行将关闭lecture和fqdn标志，并打开tty_tickets：

```
Defaults !lecture,tty_tickets,!fqdn
```

**env_reset**

使sudo来重置环境变量使之仅包含LOGNAME、SHELL、USER、USERNAME、MAIL和SUDO_*变量。默认为on。更多信息，请参阅sudoers手册页。

**fqdn**

（fully qualified domain name：完全合格域名）在sudoers文件中的FQDN上执行DNS查找。当设置了这个标志，可以在sudoers文件中使用FQDN，但这样做可能会对sudo的性能产生负面影响，特别是如果DNS不能正常工作时。当设置了这个标志，必须使用本地主机的官方DNS名称，不能用别名。如果hostname 返回一个FQDN，则不需要设置这个标志。默认是on。

**insults**

当用户输入一个错误密码时，显示温和、幽默的insults。默认是off。另见passwd_tries。

**lecture=freq**

控制sudo在密码提示之前显示提醒消息的频率。freq可能的值是never、once和always。指定!lecture与指定为never的freq是相同的。默认是once。

**mail_always**

每当用户运行sudo时，给mailto用户发送电子邮件。默认是off。

**mail_badpass**

当用户运行sudo时输入一个错误的密码时，给mailto用户发送电子邮件。默认是off。

**mail_no_host**

当一个用户的用户名在sudoers文件中，但在本地主机上没有运行命令的特权时运行sudo的，给mailto用户发送电子邮件。默认是off。

**mail_no_perms**

当用户的用户名在sudoers文件中，但没有权限运行被请求命令（运行sudo）的，给mailto用户发送电子邮件。默认是off。

**mail_no_user**

当其用户名不在sudoers文件中的用户运行sudo的，给mailto用户发送电子邮件。默认是on。

**mailsub=subj**

(mail subject）将用于警告和错误消息的默认电子邮件主题从默认的*** SECURITY information for %h ***更改为subj。sudo实用程序在subj内将%h扩展到本地系统的主机名。如果subj包含shell特殊字符，把它放在引号之间。

**mailto=eadd**

给eadd（电子邮件地址，默认是root）发送sudo的警告和错误消息。如果eadd包含shell特殊字符，把它放在引号之间。

**passwd_timeout=mins**

mins是 sudo密码超时的分钟数。值为 0（零）表示密码不会超时。默认值为 5。

**passwd_tries=num**

num是为响应sudo密码提示并在sudo退出之前，用户可以输入错误密码的次数。默认为 3。另见insults和lecture。

**rootpw**

使sudo只接受root密码以响应其提示符。因为无论是要求的密码或root密码，sudo都发布相同的提示符，开启这个标志可能会混淆用户。默认是off，使sudo提示运行sudo的用户输入密码。

**shell_noargs**

当不带任何参数调用时，会使sudo无须改变环境就派生一个root shell。默认是off。此选项与sudo的-s选项是相同的。

**timestamp_timeout=mins**

mins是sudo时间戳有效的分钟数。设置mins为-1 会使时间戳永远有效；设置为 0（零）会使sudo始终提示输入密码。默认值为 5。

**tty_tickets**

使sudo基于每个tty验证用户，而不是基于每个用户。默认是on。

**umask=val**

val是sudo用来运行该用户指定命令的umask值。将val设置为 0777，以保留用户的umask值。默认值是 0022。

### 4. 锁定 root 账户（删除 root 密码）

如果决定要锁定root账户，发出su-c'passwd-l root'命令。通过在其前面加两个惊叹号（!!），此命令将使/etc/shadow中的加密密码设置为无效。可以通过删除惊叹号来解锁该账户或发出下面示例中所示的命令。

### 5. 解锁 root 账户

如果决定解锁锁定的root账户，发出下面的命令。此命令假定可以使用sudo来取得root特权，并通过给它分配一个密码来解锁root账户：

```
$ sudo passwd root
[sudo] password for sam:
Changing password for user root.
```

```
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

### 6. 允许普通用户运行特权命令

consolehelper实用工具使得通常只能由root特权用户运行的系统程序更容易由普通用户运行。PAM对用户进行身份验证，可以设置信任所有控制台用户，要求输入用户密码（非root密码），或在授予信任之前输入root密码。consolehelper的概念基础是，可能要把对控制台有访问权限的任何人看作是值得信赖的。例如，Sam可以用自己的身份在控制台上登录，无须知道root密码就可以运行reboot。

要了解consolehelper的工作原理，可以参考这两个halt文件：

```
$ file /sbin/halt /usr/bin/halt
/sbin/halt: symbolic link to '../bin/systemctl'
/usr/bin/halt: symbolic link to 'consolehelper'
```

在Fedora下，/sbin中的文件链接到/bin/systemctl；在RHEL下则链接到/sbin/reboot。在这两个系统上，/usr/bin中的文件都是到/usr/bin/consolehelper的一个链接。这些链接文件都是可执行文件：

```
$ file /bin/systemctl /usr/bin/consolehelper
/bin/systemctl: ELF 32-bit LSB executable, Intel 80386 ...
/usr/bin/consolehelper: ELF 32-bit LSB executable, Intel 80386 ...
```

在root的PATH变量中，/sbin通常先于/usr/bin。因此，当用户以root特权运行在root登录环境中发出halt命令，shell将执行Fedora下的/bin/systemctl reboot或RHEL下的/sbin/reboot。

在一个普通用户的PATH变量中，/usr/bin通常先于/sbin（如果PATH包含/sbin）。因此，当一个普通用户发出一个reboot命令，shell将执行/usr/bin/consolehelper（consolehelper实用程序）。

consolehelper做什么取决于如何设置PAM。默认情况下，consolehelper并不需要root密码；任何用户都可以发出reboot命令来重启系统。

## 5.4 企业Linux进程安全防护

Linux是一个多用户、多任务的操作系统。在这样的系统中，各种计算机资源（如文件、内存、CPU等）的分配和管理都以进程为单位。为了协调多个进程对这些共享资源的访问，操作系统要跟踪所有进程的活动，以及它们对系统资源的使用情况，从而实施对进程和资源的动态管理。进程在一定条件下可以对诸如文件、数据库等客体进行操作。如果进程用作其他不法用途，将会给系统带来重大危害。在现实生活中，许多网络黑客都是通过种植“木马”的办法来达到破坏计算机系统和入侵的目的，而这些“木马”程序无一例外需要通过进程这一方式在机器上运行才能发挥作用。另外，许多破坏程序和攻击手段都需要通过破坏目标计算机系统的合法进程尤其是重要系统进程，使得系统不能完成正常工作甚至无法工作，从而达到摧毁目标计算机系统的目的。作为服务器中占绝大多数市场份额的Linux系统，要切实保证计算机系统的安全，我们必须对其进程进行监控和保护。

Linux操作系统包括以下三种不同类型的进程，每种进程都有其自己的特点和属性。

- 交互进程：由一个 shell 启动的进程。交互进程既可以在前台运行，也可以在后台运行。
- 批处理进程：这种进程和终端没有联系，是一个进程序列。
- 守护进程：Linux 系统启动时启动的进程，并在后台运行。

上述三种进程各有各的作用，使用场合也有所不同。

### 5.4.1 确定 Linux 下的重要进程

表 5-2 列出了Linux系统中一些比较重要的守护进程以及其所具有的功能，用户可以通过使用这些进程方便地使用系统以及网络服务。

表 5-2 Linux 重要守护进程列表

| 守护进程 | 功能说明 |
|---|---|
| acpid | acpid 是为替代传统的 APM 电源管理标准而推出的新型电源管理标准 |
| amd | 自动安装 NFS（网络文件系统） |
| apmd | 高级电源管理 |
| amd | 自动安装 NFS 守护进程 |
| anacron | 一个自动化运行任务守护进程 |
| arptables_jf | 为 arptables 网络的用户控制过滤的守护进程 |
| arpwatch | 记录日志并构建一个在 LAN 接口上看到的以太网地址和 IP 地址对数据库 |
| autofs | 自动安装管理进程 automount，与 NFS 相关，依赖于 NIS 服务器 |
| bootparamd | 引导参数服务器，为 LAN 上的无盘工作站提供引导所需的相关信息 |
| bluetooch | 蓝牙服务器守护进程 |
| crond | crond 是 UNIX 下的一个传统程序，类似计划任务 |
| chargen | 使用 TCP 协议的 chargenserver，chargen（CharacterGeneratorProtocol）是一种网络服务，主要功能是提供类似远程打字的功能 |
| chargen-udp | 使用 UDP 协议的 chargenserver |
| cpuspeed | 监测系统空闲百分比，降低或加快 CPU 时钟速度和电压，从而在系统空闲时将能源消耗降为最小，而在系统繁忙时最大化加快系统执行速度 |
| dhcpd | 启动一个 DHCP（动态 IP 地址分配）服务器 |
| cups | cups（Common UNIX Printing System）是通用 UNIX 打印守护进程，为 Linux 提供第三代打印功能 |
| daytime | 使用 TCP 协议的 Daytime 守护进程，该协议为客户机实现从远程服务器获取日期和时间的功能 |
| daytime-udp | 使用 UDP 协议的 Daytime 守护进程 |
| dc_server | 使用 SSL 安全套接字的代理服务器守护进程 |
| dc_client | 使用 SSL 安全套接字的客户端守护进程 |
| diskdump | 服务器磁盘备份守护进程 |
| echo | 服务器回显客户数据服务守护进程 |

续表

| 守护进程 | 功能说明 |
| --- | --- |
| echo-udp | 使用 UDP 协议的服务器回显客户数据服务守护进程 |
| eklogin | 接受 rlogin 会话签证和用 kerberos5 加密的一种服务的守护进程 |
| gated | 网关路由守护进程 |
| gpm | gpm（General Purpose MouseDaemon）守护进程为文本模式下的 Linux 程序如 mc（MidnightCommander）提供了鼠标的支持 |
| gssftp | 使用 kerberos5 认证的 FTP 守护进程 |
| httpd | Web 服务器 |
| inetd | 因特网操作守护程序。监控网络对各种它管理的服务的需求，并在必要的时候启动相应的服务程序 |
| innd | Usenet 新闻服务器守护进程 |
| iptables | iptables 防火墙守护进程 |
| irda | 红外端口守护进程 |
| isdn | isdn 启动和中止服务守护进程 |
| irqbalance | 对多个系统处理器环境下的系统中断请求进行负载平衡的守护程序 |
| krb5-telnet | 使用 Kerberos5 认证的 Telnet 守护进程 |
| klogin | 远程登录守护进程 |
| keytable | 该进程的功能是转载在/etc/sysconfig/keyboards 里定义的键盘映射 |
| kshell | kshell 守护进程 |
| kudzu | 硬件自动检测程序，会自动检测硬件是否发生变动，并相应进行硬件的添加、删除工作 |
| ldap | ldap（Lightweight Directory Access Protocol）目录访问协议服务器守护进程 |
| lm_seroems | 检测主板工作情况守护进程 |
| lpd | 打印服务器 |
| mdmonitor | RAID 相关设备的守护程序 |
| messagebus | D-BUS 是一个库，为两个或两个以上的应用程序提供一对一的通信 |
| microcode_ctl | 可编码以及发送新的微代码到内核以更新 IntelIA32 系列处理器守护进程 |
| mysqld | 一个快速、高效、可靠的轻型 SQL 数据库引擎守护进程 |
| named | DNS 服务器 |
| netplugd | netplugd（Network Cable Hotplug Management Daemon）守护程序，用于监控一个或多个网络接口的状态，当某些事件触发时运行一个外部脚本程序 |
| netdump | 远程网络备份服务器守护进程 |
| netfs | NetworkFilesystemMounter，该进程安装和卸载 NFS、SAMBA 和 NCP 网络文件系统 |
| nfsd | NFS 服务器 |
| nfslock | NFS 是一个流行的通过 TCP/IP 网络共享文件的协议，此守护进程提供了 NFS 文件锁定功能 |
| ntpd | Network time Protocol daemon（网络时间校正协议） |

续表

| 守护进程 | 功能说明 |
| --- | --- |
| network | 激活/关闭启动时的各个网络接口守护进程 |
| psacct | 该守护进程包括几个工具用来监控进程活动的工具，包括 ac、lastcomm、accton 和 sa |
| pcmcia | 主要用于支持笔记本电脑接口守护进程 |
| portmap | 该守护进程用来支持 RPC 连接，RPC 被用于 NFS 以及 NIS 等服务 |
| postgresql | PostgreSQL 关系数据库引擎 |
| proftpd | proftpd 是 UNIX 下的一个配置灵活的 FTP 服务器的守护程序 |
| pppoe | ADSL 连接守护进程 |
| random | 保存和恢复系统的高质量随机数生成器，这些随机数是系统一些随机行为提供的 |
| rawdevices | 在使用集群文件系统时用于加载 raw 设备的守护进程 |
| rhnsd | RedHat 网络服务守护进程。通知官方的安全信息以及为系统打补丁 |
| routed | 该守护程序支持 RIP 协议的自动 IP 路由表维护 |
| rsync | remotesync 远程数据备份守护进程 |
| rsh | 远程主机上启动一个 shell，并执行用户命令 |
| rwhod | 允许远程用户获得运行 rwho 守护程序的机器上所有已登录用户的列表 |
| rstatd | 一个为 LAN 上的其他机器收集和提供系统信息的守候进程 |
| ruserd | 远程用户定位服务，这是一个基于 RPC 的服务，它提供关于当前记录到 LAN 上一个机器日志中的用户信息 |
| rwalld | 激活 rpc.rwall 服务进程，这是一项基于 RPC 的服务，允许用户给每个注册到 LAN 机器上的其他终端写消息 |
| saslauthd | 使用 SASL 的认证守护进程 |
| sendmail | 邮件服务器 sendmail |
| smb | Samba 文件共享/打印服务 |
| snmpd | 本地简单网络管理守护进程 |
| squid | 代理服务器 squid 守护进程 |
| sshd | OpenSSH 服务器守护进程。Secure Shell Protocol 可以实现安全地远程管理主机 |
| smartd | SelfMonitor Analysisand Reporting Technology System，监控你的硬盘是否出现故障 |
| syslog | 一个让系统引导时启动 syslog 和 klogd 系统日志守候进程的脚本 |
| time | 该守护进程从远程主机获取时间和日期，采用 TCP 协议 |
| time-udp | 该守护进程从远程主机获取时间和日期，采用 UDP 协议 |
| tux | 在 Linux 内核中运行 Apache 服务器的守护进程 |
| vsftpd | vsftpd 服务器的守护进程 |
| vncserver | VNC（Virtual Network Computing，虚拟网络计算），它提供了一种在本地系统上显示远程计算机整个“桌面”的轻量型协议 |
| xfs | XWindow 字型服务器守护进程，为本地和远程 X 服务器提供字型 |
| xinetd | 支持多种网络服务的核心守候程序 |
| ypbind | 为 NIS（网络信息系统）客户机激活 ypbind 服务进程 |

续表

| 守护进程 | 功能说明 |
| --- | --- |
| yppasswdd | NIS 口令服务器守护进程 |
| ypserv | NIS 主服务器守护进程 |
| yum | RPM 操作系统自动升级和软件包管理守护进程 |

### 5.4.2 进程安全命令行管理方法

Linux系统提供了who、w、ps和top等查看进程信息的系统调用，通过结合使用这些系统调用，我们可以清晰地了解进程的运行状态以及存活情况，从而采取相应的措施，来确保Linux系统的安全。

（1）who命令：该命令主要用于查看当前在线上的用户情况。系统管理员可以使用who命令监视每个登录的用户此时此刻的所作所为：

```
# who
root     pts/1        2010-02-21 15:56 (:0.0)
```

（2）w命令：该命令也用于显示登录到系统的用户情况，但是与who不同的是，w命令功能更加强大，它不但可以显示有谁登录到系统，还可以显示出这些用户当前正在进行的工作。w命令是who命令的一个增强版：

```
# w
 15:56:44 up 38 min,  1 user,  load average: 0.56, 0.15, 0.12
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
root     pts/1    :0.0             15:56    0.00s  0.19s  0.05s w
```

（3）ps命令：是最基本同时也是非常强大的进程查看命令。使用该命令可以确定有哪些进程正在运行和运行的状态、进程是否结束、进程有没有僵死、哪些进程占用了过多的资源等。ps命令可以监控后台进程的工作情况，因为后台进程是不和屏幕键盘这些标准输入/输出设备进行通信的，如果需要检测其情况，可以使用ps命令。下面是一个ps命令的例子：

```
# ps
  PID TTY          TIME CMD
 2817 pts/1    00:00:00 bash
 2836 pts/1    00:00:00 ps
```

（4）top 命令：top 命令和 ps 命令的基本作用是相同的，显示系统当前的进程和其他状况；但是 top 是一个动态显示过程，可以通过用户按键来不断刷新当前状态。如果在前台执行该命令，它将独占前台，直到用户终止该程序为止。比较准确地说，top 命令提供了实时地对系统处理器的状态监视。它将显示系统中 CPU 最“敏感”的任务列表。该命令可以按 CPU 使用。内存使用和执行时间对任务进行排序；而且该命令的很多特性都可以通过交互式命令或者在个人定制文件中进行设定。下面是一个 top 命令的例子：

```
# top
top - 15:58:07 up 39 min,  1 user,  load average: 2.09, 0.68, 0.30
Tasks: 112 total,   2 running, 108 sleeping,   0 stopped,   2 zombie
Cpu(s): 30.6%us, 25.2%sy, 0.0%ni, 41.2%id,  0.3%wa, 2.7%hi, 0.0%si, 0.0%st
```

```
Mem:   485736k total,  477828k used,    7908k free,   31252k buffers
Swap: 1285160k total,       0k used, 1285160k free,  291192k cached
 PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 2502 root      15   0 36680  10m 5776 S 40.0  2.2   0:54.26 Xorg
 2814 root      15   0  127m  16m  10m S  3.0  3.5   0:03.62 gnome-terminal
 2616 root      15   0 97356  14m  10m S  2.6  3.1   0:02.43 gnome-panel
 2609 root      15   0 38660  10m 7544 S  2.0  2.1   0:01.57 metacity
 2640 root      15   0 96276  13m 9448 S  2.0  2.7   0:01.46 wnck-applet
 2597 root      15   0 33916 8144 6768 S  1.0  1.7   0:01.32 gnome-settings-
 2618 root      15   0  133m  20m  14m S  1.0  4.4   0:04.66 nautilus
 2709 root      15   0 16396 3488 2736 S  0.7  0.7   0:02.72 gnome-screensav
 2837 root      15   0  2160 1008  788 R  0.7  0.2   0:00.31 top
 2855 root      15   0  106m  22m  14m S  0.7  4.8   0:01.87 gedit
 2035 root      18   0 33180 1276  536 S  0.3  0.3   0:01.51 pcscd
 2076 root      21   0  9336 1108  856 S  0.3  0.2   0:00.35 automount
 2313 root      18   0  1920  624  544 S  0.3  0.1   0:03.46 hald-addon-stor
 2665 root      15   0  2528 1156  948 S  0.3  0.2   0:00.26 gam_server
 2703 root      15   0 64020  24m  14m S  0.3  5.3   0:02.39 /usr/bin/sealer
 2713 root      15   0 49500 6528 3484 S  0.3  1.3   0:00.31 scim-panel-gtk
    1 root      15   0  2036  640  548 S  0.0  0.1   0:02.16 init
    2 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    3 root      34  19     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/0
    4 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 watchdog/0
    5 root      10  -5     0    0    0 S  0.0  0.0   0:00.06 events/0
    6 root      10  -5     0    0    0 S  0.0  0.0   0:00.01 khelper
    7 root      10  -5     0    0    0 S  0.0  0.0   0:00.01 kthread
   10 root      10  -5     0    0    0 S  0.0  0.0   0:00.26 kblockd/0
```

以上介绍的是目前在Linux下使用最常见的进程状况查看工具，它们是随Linux套件发行的，安装好系统之后，用户就可以使用。当然，随着开源的不断发展，相信将会有更多的该方面的工具出现，以方便用户选择和使用。

### 5.4.3 使用进程文件系统管理进程

顾名思义，PROC文件系统是一个虚拟的文件系统，通过文件系统的接口实现，用于输出系统的运行状态。它以文件系统的形式，为操作系统本身和应用进程之间的通信提供了一个界面，使应用程序能够安全、方便地获得系统当前的运行状况和内核的内部数据信息，并可以修改某些系统的配置信息。另外，由于PROC以文件系统的接口实现，因此用户可以像访问普通文件一样对其进行访问，但它只存在于内存之中，并不存在于真正的物理磁盘当中。所以，当系统重启和电源关闭的时候，该系统中的数据和信息将全部消失。

表 5-3 说明了该文件系统中一些重要的文件和目录。

表 5-3　重要的 PROC 文件或目录

| 文件或目录 | 说　　明 |
|---|---|
| /proc/1 | 关于进程 1 的信息目录。每个进程在/proc 下有一个名为其进程号的目录 |
| /proc/cpuinfo | 处理器信息，如类型、制造商、型号和性能 |
| /proc/devices | 当前运行的核心配置的设备驱动的列表 |
| /proc/dma | 显示当前使用的 DMA 通道 |
| /proc/filesystems | 核心配置的文件系统 |
| /proc/interrupts | 显示使用的中断 |
| /proc/ioports | 当前使用的 I/O 端口 |
| /proc/kcore | 系统物理内存映像 |
| /proc/kmsg | 核心输出的消息，也被送到 syslog |
| /proc/ksyms | 核心符号表 |
| /proc/loadavg | 系统的平均负载 |
| /proc/meminfo | 存储器使用信息，包括物理内存和 swap |
| /proc/modules | 当前加载了哪些核心模块 |
| /proc/net | 网络协议状态信息 |
| /proc/stat | 系统的不同状态 |
| /proc/version | 核心版本 |
| /proc/uptime | 系统启动的时间长度 |
| /proc/cmdline | 命令行参数 |

下面将通过一个例子来说明，如何使用PROC文件系统来获得进程的信息。

（1）首先使用vi编辑器建立一个C源程序文件，编译后形成目标文件，该文件的主要功能是进行计算（如何在Linux下编写C程序不是本书强调的内容，请读者参看相关的技术书籍和文档），并保存在/home目录下，将其运行。使用ps-ef命令，则能发现在系统中运行了proc_test这样一个进程，如图 5-4 所示。

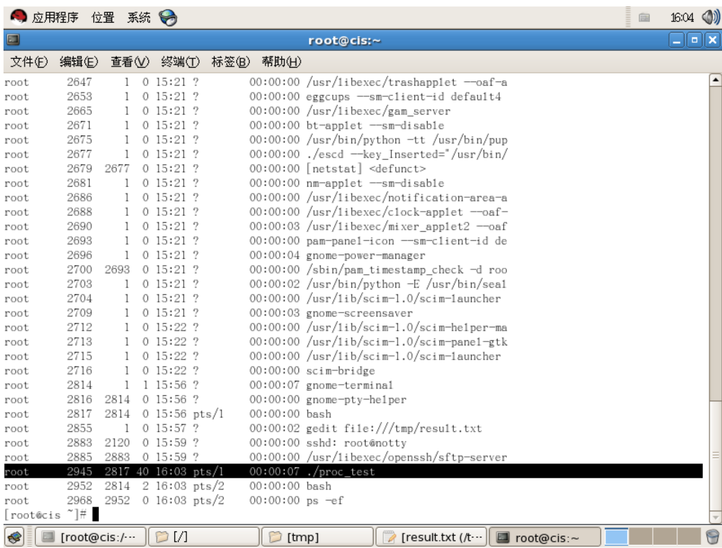



图 5-4　显示 proc_test 进程

（2）进程的基本信息都会存放在/proc文件系统中，具体位置是在/proc目录下。通过使用如下命令可以查看系统中运行进程的相关信息，如图 5-5 所示，其中显示为系统中运行进程的信息所存放的目录，每个进程对应一个目录，2945 为例子使用的进程的详细信息所在目录。

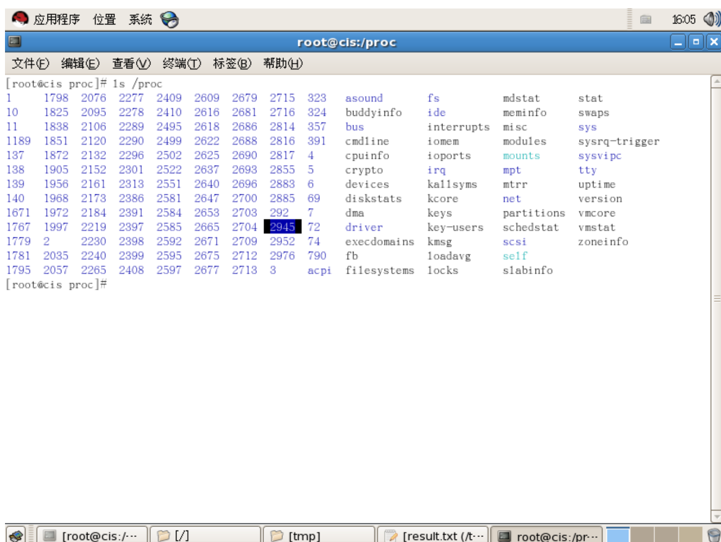



图 5-5　/ls/proc 命令显示结果

（3）切换到 2945 目录，以方便详细的查看进程信息，并列出进程详细地状态信息文件，如图 5-6 所示。

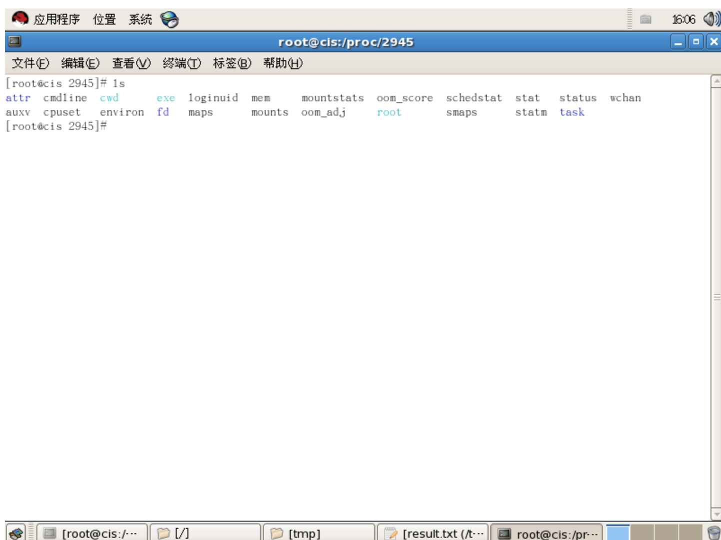



图 5-6　进程 2945 信息所在目录

（4）在这些文件当中，status这个状态文件是比较重要的，包含了很多关于进程的有用的信息，用户可以从这个文件获得信息，如下所示。

```
Name:   proc_test
State:  R (running)
SleepAVG:       0%
Tgid:   2945
Pid:    2945
PPid:   2817
TracerPid:      0
Uid:    0       0       0       0
Gid:    0       0       0       0
FDSize: 256
```

```
Groups: 0 1 2 3 4 6 10
VmPeak:     1564 kB
VmSize:     1484 kB
VmLck:         0 kB
VmHWM:       256 kB
VmRSS:       248 kB
VmData:       24 kB
VmStk:        88 kB
VmExe:         4 kB
VmLib:      1344 kB
VmPTE:        20 kB
StaBrk: 0804a000 kB
Brk:    090eb000 kB
StaStk: bfdf4ea0 kB
ExecLim:       08049000
Threads:       1
SigQ:   2/9792
SigPnd: 0000000000000000
ShdPnd: 0000000000000000
SigBlk: 0000000000000000
SigIgn: 0000000000000000
SigCgt: 0000000000000000
CapInh: 0000000000000000
CapPrm: 00000000fffffeff
CapEff: 00000000fffffeff
Cpus_allowed:   ffffffff
Mems_allowed:   1
```

其中，比较重要的字段详细含义如下：

```
Name:   proc_test                                          //进程名
State:  R (running)                                        //进程运行状态
Tgid:   2945                                               //进程组 ID
Pid:    2945                                               //进程 ID
PPid:   2817                                               //父进程 ID
TracerPid:     0                                           //跟踪调试进程 ID
Uid:    0    0    0    0                                   //进程所对应程序的 UID
Gid:    0    0    0    0                                   //进程所对应程序的 GID
FDSize: 256                                                //进程使用文件句柄大小
```

### 5.4.4 管理中常用的 PROC 文件系统调用接口

LKM程序员也可以使用标准的API来使用PROC文件系统。LKM甚至可以导出内核使用的新变量和函数。有关API的完整介绍不是本书所要叙述的主要内容，因此在下面我们只是简单地介绍一些常用的API，供读者在实际中选择使用。

1．创建并删除/proc 项

要在/proc文件系统中创建一个虚拟文件，需使用create_proc_entry函数。该函数可以接收一个文件名、一组权限和这个文件在/proc文件系统中出现的位置。create_proc_entry的返回值是一个proc_dir_entry指针（或者为NULL，说明在create时发生了错误）。然后就可以使用这个返回的指针来配置这个虚拟文件的其他参数，例如在对该文件执行读操作时应该调用的函数。create_proc_entry的原型和proc_dir_entry结构中的一部分程序如下：

```
//有关 PROC 文件系统项的数据结构
struct proc_dir_entry *create_proc_entry( const char *name, mode_t mode,
                                      struct proc_dir_entry *parent );
struct proc_dir_entry {
const char *name;                         // virtual file name
mode_t mode;                              // mode permissions
uid_t uid;                                // File's user id
gid_t gid;                                // File's group id
struct inode_operations *proc_iops;       // Inode operations functions
struct file_operations *proc_fops;        // File operations functions
struct proc_dir_entry *parent;            // Parent directory
...
//PROC 文件系统的读函数
read_proc_t *read_proc;                   // /proc read function
//PROC 文件系统的写函数
write_proc_t *write_proc;                 // /proc write function
void *data;                               // Pointer to private data
atomic_t count;                           // use count
...
};
//删除 PROC 项的函数
void remove_proc_entry( const char *name, struct proc_dir_entry *parent );
```

在后面的例子中我们将介绍如何使用read_proc和write_proc命令来插入对这个虚拟文件进行读写的函数。要从/proc中删除一个文件，可以使用remove_proc_entry函数。要使用这个函数，我们需要提供文件名字符串，以及该文件在/proc文件系统中的位置（parent）。parent参数可以为NULL（表示/proc根目录），也可以是很多其他值，这取决于我们希望将该文件放到什么地方。表 5-4 列出了可以使用的其他一些父proc_dir_entry，以及它们在文件系统中的位置。

表 5-4　一些常用的 proc_dir_entry 快捷变量

| 名　称 | 目　录 |
| --- | --- |
| proc_root_fs | /proc |
| proc_net | /proc/net |
| proc_bus | /proc/bus |
| proc_root_driver | /proc/driver |

### 2. 回调函数

（1）写回调函数。

我们可以使用write_proc函数向/proc中写入一项。这个函数的原型如下：

```
int mod_write( struct file *filp, const char __user *buff,
               unsigned long len, void *data );
```

filp参数实际上是一个打开文件结构（我们可以忽略这个参数）。buff参数是传递给用户的字符串数据。缓冲区地址实际上是一个用户空间的缓冲区，因此我们不能直接读取它。len参数定义了在buff中有多少数据要被写入。data参数是一个指向私有数据的指针。在这个模块中，我们声明了一个这种类型的函数来处理到达的数据。Linux提供了一组API来在用户空间和内核空间之间移动数据。对于write_proc的情况来说，我们使用了copy_from_user函数来维护用户空间的数据。

（2）读回调函数。

我们可以使用read_proc函数从一个/proc项中读取数据（从内核空间到用户空间）。这个函数的原型如下：

```
int mod_read( char *page, char **start, off_t off,
              int count, int *eof, void *data );
```

page参数是这些数据写入到的位置，其中count定义了可以写入的最大字符数。在返回多页数据（通常一页是 4KB）时，我们需要使用start和off参数。当所有数据全部写入之后，就需要设置eof（文件结束参数）。与write类似，data表示的也是私有数据。此处提供的page 缓冲区在内核空间中。因此，我们可以直接写入，而不用调用copy_to_user。

（3）其他有用的函数。

我们还可以使用proc_mkdir、symlinks以及proc_symlink在/proc文件系统中创建目录。对于只需要一个read函数的简单/proc项来说，可以使用create_proc_read_entry，这样会创建一个/proc项，并在一个调用中对read_proc函数进行初始化。这些函数的原型如下所示：

```
//创建一个目录
struct proc_dir_entry *proc_mkdir( const char *name,
                               struct proc_dir_entry *parent );


//创建一个符号链接
struct proc_dir_entry *proc_symlink( const char *name,
                                 struct proc_dir_entry *parent,
                                 const char *dest );
```

```
以读函数创建一个 proc 目录项
struct proc_dir_entry *create_proc_read_entry( const char *name,
                                          mode_t mode,
                                          struct proc_dir_entry *base,
                                          read_proc_t *read_proc,
                                          void *data );

//用于从内核空间拷贝数据到用户空间的函数
unsigned long copy_to_user( void __user *to,
                         const void *from,
                         unsigned long n );

//用于从用户空间拷贝数据到内核空间的函数
unsigned long copy_from_user( void *to,
                           const void __user *from,
                           unsigned long n );

//虚拟内存分配函数
void *vmalloc( unsigned long size );

//虚拟内存释放函数
/* Free a vmalloc'd block of memory */
void vfree( void *addr );

//将符号导入到内核
EXPORT_SYMBOL( symbol );

//将一个文件的所有符号导入到内核
EXPORT_SYMTAB
```

## 5.5 企业 Linux 用户安全管理

### 5.5.1 管理用户及组文件安全

Linux操作系统采用了UNIX传统的方法，把全部的用户信息保存为普通的文本文件。用户可以通过对这些文件进行修改来管理用户和组。本节将对这些文件的结构进行详细介绍。

### 1. 用户账号文件——passwd

/etc/passwd文件是UNIX安全的关键文件之一。该文件用于用户登录时校验用户的登录名、加密的口令数据项、用户ID（UID）、默认的用户分组ID（GID）、用户信息、用户登录子目录以及登录后使用的shell。这个文件的每一行保存一个用户的资料，而用户资料的每一个数据项采用冒号“：”分隔。如下所示：

```
LOGNAME：PASSWORD：UID：GID：USERINFO：HOME：SHELL
```

每行的头两项是登录名和加密后的口令，后面的两个数是UID和GID，接着的一项是系统管理员想写入的有关该用户的任何信息，最后两项是两个路径名：一个是分配给用户的HOME目录，另一个是用户登录后将执行的shell（若为空格，则默认为/bin/sh）。

下面是一个实际的系统用户的例子：

```
cracker:x:6018: 6018: cracker:/home/ cracker:/bin/bash
```

该用户的基本信息如下。

- 登录名：cracker。
- 加密的口令表示：x。
- UID：6018。
- GID：6018。
- 用户信息：cracker。
- HOME 目录：/home/ cracker。
- 登录后执行的 shell：/bin/bash。

用户的登录名是用户用来登录的识别，由用户自行选定，主要由方便用户记忆或者具有一定含义的字符串组成。

所有用户口令的存放都是加密的，通常采用的是不可逆的加密算法，比如DES（Data Encryption Standard，数据加密标准）。当用户在登录提示符处输入它们的口令时，输入的口令将由系统进行加密，再把加密后的数据与机器中用户的口令数据项进行比较。如果这两个加密数据匹配，就可以让这个用户进入系统。在/etc/passwd文件中，UID信息也很重要。系统使用UID而不是登录名区别用户。一般来说，用户的UID应当是独一无二的，其他用户不应当有相同的UID数值，只有UID=0 时可以例外。任何拥有 0 值UID的用户都具有根用户（系统管理员）访问权限，因此具备对系统的完全控制。通常，UID为 0 这个特殊值的用户的登录名是“root”。根据惯例，从 0～99 的UID保留用作系统用户的UID。如果在/etc/passwd文件中有两个不同的入口项有相同的UID，则这两个用户对文件具有相同的存取权限。

每一个用户都需要有地方保存专属于自己的配置文件。这需要让用户工作在自己定制的操作环境中，以免改变其他用户定制的操作环境，这个地方就叫作用户登录子目录。在这个子目录中，用户不仅可以保存自己的配置文件，还可以保存自己日常工作中用到的各种文件。出于一致性的考虑，大多数站点都从/home开始安排用户登录子目录，并把每个用户的子目录命名为其上机使用的登录名。

当用户登录进入系统时，都有一个属于自己的操作环境。用户遇到的第一个程序叫作shell。在Linux系统里，大多数shell都是基于文本的。Linux操作系统带有多种shell供用户选用。用户可以在/etc/shells文件中看到它们的绝大多数。用户可以根据自己的喜好来选用不同的shell进行操作。按照最严格的定义，在上面所介绍的/etc/passwd文件中，每个用户的口令数据项中并没

有定义需要运行某个特定的shell，其中列出的是这个用户上机后第一个运行的程序是哪个。综上所述，通过使用 cat 命令查看/etc/passwd 文件（#cat /etc/passwd），可以得到如图 5-7 所示的完整的系统账号文件：

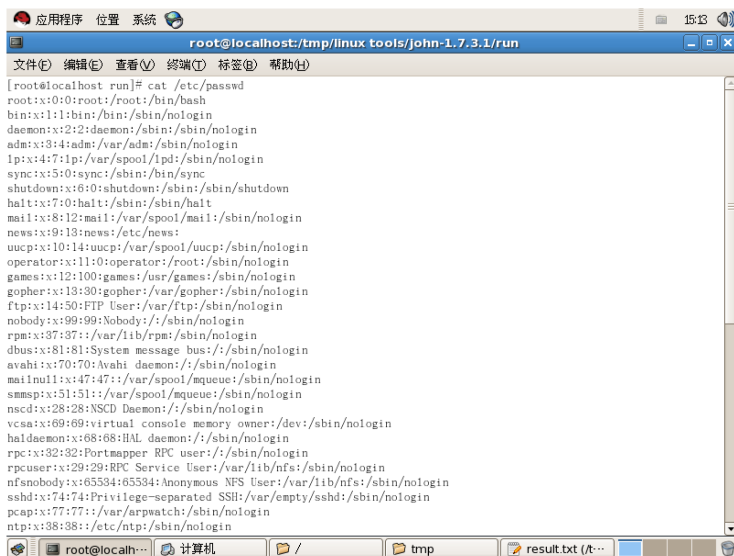



图 5-7 /etc/passwd 文件

### 2. 用户影子文件——shadow

Linux使用不可逆的加密算法如DES来加密口令，由于加密算法是不可逆的，所以黑客从密文是得不到明文的。但/etc/passwd文件是全局可读的，加密的算法是公开的，恶意用户取得了/etc/passwd文件，便极有可能破解口令。而且，在计算机性能日益提高的今天，对账号文件进行字典攻击的成功率会越来越高，速度越来越快。因此，针对这种安全问题，Linux/UNIX广泛采用了"shadow（影子）文件"机制，将加密的口令转移到/etc/shadow文件里，该文件只为root超级用户可读，而同时/etc/passwd文件的密文域显示为一个x，从而最大限度地减少了密文泄露的机会。

/etc/shadow文件的每行是 8 个冒号分隔的 9 个域，格式如下：

```
username: passwd: lastchg: min: max: warn: inactive: expire: flag
```

其中，各个域的含义如表 5-5 所示：

表 5-5 /etc/shadow 文件域名含义

| 域　名 | 含　义 |
|---|---|
| username | 用户登录名 |
| passwd | 加密的用户口令 |
| lastchg | 表示从 1970 年 1 月 1 日起到上次修改口令所经过的天数 |
| min | 表示两次修改口令之间至少经过的天数 |
| max | 表示口令还会有效的最大天数，如果是 99999 则表示永不过期 |
| warn | 表示口令失效前多少天内系统向用户发出警告 |
| inactive | 表示禁止登录前用户名还有效的天数 |
| expire | 表示用户被禁止登录的时间 |
| flag | 保留域，暂未使用 |

如下所示的是一个系统中实际影子文件的例子：

```
liyang:$1$ciY58zQZ$iKVHLSVZZgM75.lGp5Rmv.:14633:0:99999:7:::
```

我们对最后一个用户的信息进行解释，该信息表明了如下含义。

- 用户登录名：liyang。

- 用户加密的口令：liyang 后紧跟的一段乱码信息$1$ciY58zQZ$iKVHLSVZZgM75.lGp5Rmv.。
- 从 1970 年 1 月 1 日起到上次修改口令所经过的天数为：14633 天。
- 需要多少天才能修改这个命令：0 天。
- 该口令永不过期：采用 99999 表示。
- 要在口令失效前 7 天通知用户，发出警告。
- 禁止登录前用户名还有效的天数未定义，以“：”表示。
- 用户被禁止登录的时间未定义，以“：”表示。
- 保留域，未使用，以“：”表示。

### 3. 组账号文件——group

/etc/passwd文件中包含着每个用户默认的分组ID（GID）。在/etc/group文件中，这个GID被映射到该用户分组的名称以及同一分组中的其他成员去。

/etc/group文件含有关于小组的信息，/etc/passwd中的每个GID在文件中应当有相应的入口项，入口项中列出了小组名和小组中的用户，这样可方便地了解每个小组的用户，否则必须根据GID在/etc/passwd文件中从头至尾地寻找同组用户，这提供了一个比较快捷的寻找途径。/etc/group文件对小组的许可权限的控制并不是必要的，因为系统用来自于/etc/passwd文件的UID、GID来决定文件存取权限，即使/etc/group文件不存在于系统中，具有相同的GID用户也可以小组的存取许可权限共享文件。小组就像登录用户一样可以有口令。如果/etc/group文件入口项的第二个域为非空（通常用x表示），则将被认为是加密口令。/etc/group文件中每一行的内容如下。

- 用户分组名。
- 加过密的用户分组口令。
- 用户分组 ID 号（GID）。
- 以逗号分隔的成员用户清单。

以下是系统中一个具体的/etc/group文件中记录的例子：

```
adm:x:4:root,adm,daemon
```

以上面文件第四行为例子，它说明在系统存在一个adm的用户组，它的信息如下：

- 用户分组名为 adm。
- 用户组口令已经加密，用“x”表示。
- GID 为 4。
- 同组的成员用户有：root、adm、daemon。

### 4. 组账号文件——gshadow

如同用户账号文件的作用一样，组账号文件也是为了加强组口令的安全性，防止黑客对其实行的暴力攻击，而采用的一种将组口令与组的其他信息相分离的安全机制。其内容如下。

- 用户组名。
- 加密的组口令。
- 组成员列表。

下面是系统中一个具体的/etc/gshadow文件的例子：

```
mail:::mail,postfix,exim
```

以组mail为例，其加密后的组口令被隐藏，其组成员包括mail、postfix和exim。其他的以"：："结尾的组表明没有组成员，但是用户可以自行添加。

### 5. /etc/skel 目录

/etc/skel目录（见图 5-8）一般是存放用户启动文件的目录，这个目录是由root权限控制。当我们添加用户时，该目录下的文件自动复制到新添加的用户的主目录下；/etc/skel目录下的文件都是隐藏文件，也就是类似.file格式的；用户可通过修改、添加、删除/etc/skel目录下的文件，来为用户提供一个统一、标准、默认的用户环境。

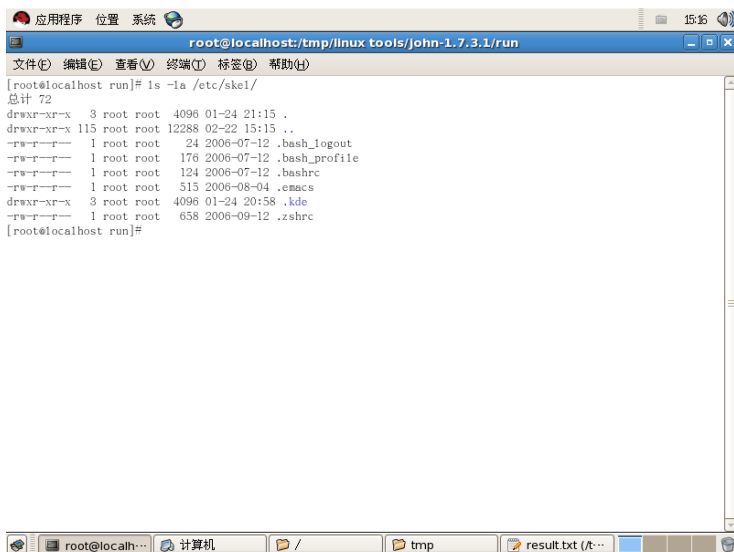


图 5-8 /etc/skel 目录

/etc/skel目录下的文件，一般是用户用useradd和adduser命令添加用户（user）时，系统自动复制到新添加用户（user）的主目录下；如果用户通过修改/etc/passwd来添加用户时，可以自己创建用户的主目录，然后把/etc/skel下的文件复制到用户的主目录下，并要用chown来改变新用户主目录的属主。

### 6. /etc/login.defs 配置文件

/etc/login.defs文件是当创建用户时的一些规划，比如创建用户时，是否需要主目录；UID和GID的范围；用户的期限等等，这个文件是可以通过root来定义的；Fedora 10下的 /etc/login.defs文件的部分内容如图 5-9 所示。

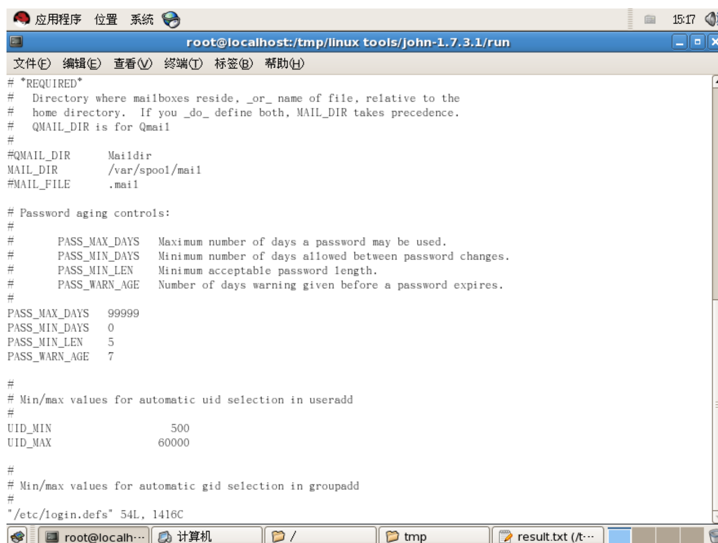


图 5-9 /etc/login.defs 文件

### 7. /etc/default/useradd 文件

通过useradd添加用户时的规则文件，如图5-10所示。

图5-10中各项的含义如下。

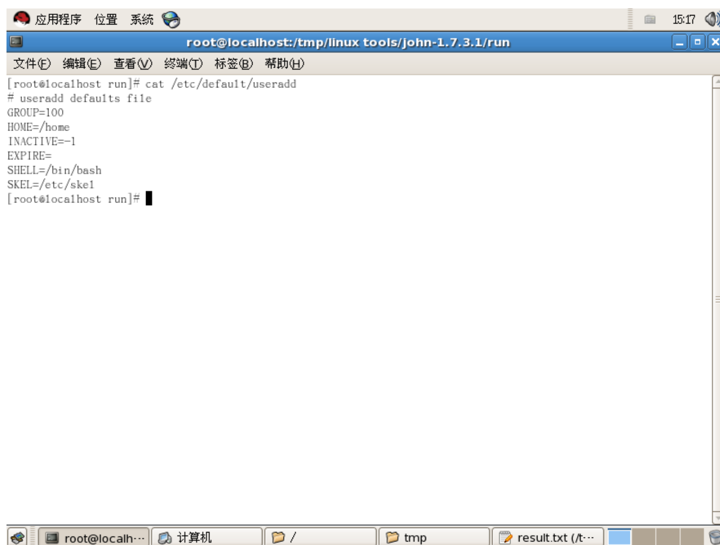



图5-10 /etc/default/useradd 文件

- GROUP=100：用户组ID。
- HOME=/home：把用户的主目录建在/home中。
- INACTIVE=-1：是否启用账号过期停权，-1表示不启用。
- EXPIRE=：账号终止日期，不设置表示不启用。
- SHELL=/bin/bash：所用SHELL的类型。
- SKEL=/etc/skel：默认添加用户的目录、默认文件存放位置；也就是说，当用户用adduser添加用户时，用户主目录下的文件，都是从这个目录中复制过去的。

## 5.5.2 用户密码管理

### 1. 密码策略的必要性

密码是用户登录Linux系统的钥匙，如果没有钥匙总是要费一番力气后，才能登录到目标操作系统。无论入侵者采用何种远程攻击，如果无法获得管理员或超级管理员的用户密码，就无法完全控制整个系统。若想访问系统，最简单也是必要的方法就是窃取用户的密码。因此，对系统管理员账户来说，最需要保护的就是密码，如果密码被盗，也就意味着灾难的降临。

入侵者大多是通过各种系统和设置漏洞，获得管理员密码来获得管理员权限的，然后，再实现对系统的恶意攻击。账号的弱密码设置会使入侵者易于破解而得以访问计算机和网络，而强密码则难以破解，即使是密码破解软件也难以在短时间内办到。密码破解软件一般使用3种方法进行破解：字典猜解、组合猜解和暴力猜解。毫无疑问，破解强密码远比破解弱密码困难得多。因此，系统管理员账户必须使用强密码。

据统计，大约80%的安全隐患是由于密码设置不当引起的。因此，密码的设置无疑是十分讲究技巧的。在设置密码时，请遵守密码安全设置原则，该原则适用于任何使用密码的场合，既包括Windows操作系统，也包括UNIX/Linux操作系统。

### 2. 密码的设置原则

（1）不可让账号与密码相同。

如果将密码设置为与用户账号相同的话，那么几乎所有的密码破解软件都将轻而易举地将密码探测出来。

（2）不可使用自己的姓名。

使用自己的姓或名，甚至是姓名作为密码，实在是不堪一击。对于本单位和熟悉本单位的人来讲，姓名无疑是攻击的首选，因为几乎谁都能猜得到。另外，在许多入侵者编写的密码猜解字典中，往往将百家姓一一列出，并放在字典的前列。

（3）不可使用英文词组。

一些常用或别致的英文单词往往是用户设置密码时的最爱。在他们看来，这类密码既便于记忆，又突显自己的个性。但事实上，那些绝顶聪明的入侵者也早已猜到并详细地将其编入密码猜解字典之中，因此，常用英文词组绝不可用作密码。

（4）不可使用特定意义的日期。

以具有特定意义的日期作为密码是任何人都十分喜爱的。这一类日期通常有自己的生日、父母的生日、儿女的生日、朋友的生日、重大节日以及个人纪念日等。不用说熟悉的人可以猜得到，即使是陌生人也可以通过穷举的方式而得手。在入侵者的密码猜解字典中，几乎全部罗列以上所有的几个组合。

（5）不可使用简单的密码。

一个密码暴力猜解软件每秒钟可以尝试 10 万次之多。字数越少，字符越简单化，排列组合的结果就越少，也就越容易被攻破。

综上所述，若要保证密码的安全，应当遵循以下规则：

- 用户密码应包含英文字母的大小写、数字、可打印字符，甚至是非打印字符。建议将这些符号排列组合使用，以期达到最好的保密效果。
- 用户密码不要太规则，不要使用用户姓名、生日、电话号码以及常用单词作为密码。
- 根据 Windows 系统密码的散列算法原理，密码长度设置应超过 7 位，最好为 14 位。
- 密码不得以明文方式存放在系统中，确保密码以加密的形式写在硬盘上并包含密码的文件是只读的。
- 密码应定期修改，应避免重复使用旧密码，并采用多套密码的命名规则。
- 建立账号锁定机制。一旦同一账号密码校验错误若干次，即断开连接并锁定该账号，经过一段时间才解锁。

密码策略包含以下 6 个策略。

- 密码必须符合复杂性要求。
- 密码长度最小值。
- 密码最长使用期限。
- 密码最短使用期限。
- 强制密码历史。
- 用可还原的加密来储存密码。

### 3. 使用密码攻击工具检查密码安全性

1）John the Ripper简介

John the Ripper是一个工具软件，用于在已知密文的情况下尝试破解出明文的破解密码软件。目前的最新版本是 1.7.9 版，主要支持对DES、MD5 两种加密方式的密文进行破解工作。它可以工作于多种不同的机型以及多种不同的操作系统之下，目前已经测试过能够正常运行的操作系统有：Linux x86、freeBSD、x86、Solaris、SPARC、OSF/1 Alpha、DOS、WinNT/WinXP

系列等。John the Ripper 1.7是目前比较好的破解密码工具，在解密过程中会自动定时存盘，用户可以强迫中断解密过程（使用Ctrl+C组合键），下次还可以从中断的地方继续进行下去（john-restore命令）。任何时候敲击键盘，用户都可以看到整个解密的进行情况，所有已经被破解的密码会被保存在当前目录下的JOHN.POT文件中，SHADOW中所有密文相同的用户会被归成一类，这样JOHN就不会进行无谓的重复劳动了。在程序的设计中，关键的密码生成的条件被放在JOHN.INI文件中，用户可以自行修改设置，不仅支持单词类型的变化，而且支持自己编写C的小程序限制密码的取值方式。

2）安装John the Ripper

在使用该软件前，我们可以从网上下载其最新版本john-1.7.9 for Linux版本，它包含DOC、SRC和RUN三个目录，在SRC目录下，在机器上执行如下命令即可：

```
#make
#make clean linux-x86-any
```

安装好后，可以切换到RUN目录下，进行测试，如下所示：

```
#cd ../run
#./john -test
```

3）使用基本命令和实用工具

（1）基本命令。

John the Ripper提供了如下多达10余种的命令，供用户选择使用。

- -pwfile:<file>[,..]：用于指定存放密文所在的文件名，（可以输入多个，文件名以“，”分隔，也可以使用*或者这两个通配符引用一批文件）。也可以不使用此参数，将文件名放在命令行的最后即可。
- -wordfile：<字典文件名>-stdin：指定的用于解密用的字典文件名。也可以使用STDIO来输入，就是在键盘中输入。
- -rules：在解密过程中使用单词规则变化功能。如将尝试cool单词的其他可能，如COOLER、Cool等，详细规则可以在JOHN.INI文件中的[List.Rules:Wordlist]部分查到。
- -incremental[:<模式名称>]：使用遍历模式，就是组合密码的所有可能情况，同样可以在JOHN.INI文件中的[Incremental:*****]部分查到。
- -single：使用单一模式进行解密，主要是根据用户名产生变化来猜测解密，可以消灭比较低级的用户。其组合规则可以在JOHN.INI文件中的[List.Rules:Single]部分查到，我们在下面将详细解释。
- -external:<模式名称>：使用自定义的扩展解密模式，用户可以在john.ini中定义自己需要的密码组合方式。JOHN也在INI文件中给出了几个示例，在INI文件的[List.External:******]中所定义的自订破解功能。
- -restore[:<文件名>]：继续上次的破解工作，JOHN被中断后，当前的解密进度情况被存放在RESTORE文件中，用户可以拷贝这个文件到一个新的文件中。如果参数后不带文件名，JOHN默认使用RESTORE文件。
- -makechars:<文件名>：制作一个字符表，用户所指定的文件如果存在，则将会被覆盖。JOHN尝试使用内在规则在相应密钥空间中生成一个最有可能击中的密码组合，它会参考在JOHN.POT文件中已经存在的密钥。

- -show：显示已经破解出的密码，因为 JOHN.POT 文件中并不包含用户名，同时用户应该输入相应的包含密码的文件名，JOHN 会输出已经被解密的用户连同密码的详细表格。
- -test：测试当前机器运行 JOHN 的解密速度，需要 1 分钟，它会得出在当前的情况下解密的各种可能情况下相应的解密速度，如同时解密 100 个用户时的平均速度，使用遍历法解密模式时解密的速度。salts 指用户个数，如果给出的对于 100 个用户解密的平均速度为 18000 次/秒，那么表明同时对 100 个用户解密，解密的速度为每个 180 次/秒。因为绝大多数的时间被用于密钥比较过程中了。所以应该对用户进行挑选。
- -users:<login|uid>[,..]：只破解某类型的用户或者属于某个组的用户。如果得到的 PASSWD 文件没有包含密文，那么在得到 SHADOW 后应该进行组合，JOHN 的附带程序 UNSHADOW.EXE 可以完成这一过程，当然了，用户也可以手工做。一般的能够进入 CSH 的用户都是解密的首选对象。也可以要 UID=0 的 ROOT 级别的用户。
- -shells:[!]<shell>[,..]：和上面的参数一样，这一选项可以选择对所有可以使用 shell 的用户进行解密，对其他用户不予理睬。“！”就是表示不要某些类型的用户。例如：“-shells：csh”。
- -salts:[!]<count>：只选择解密用户大于<count>的账号，可以使用户得到选择的权力，尽快地得到所需要的用户的 PASS。
- -lamesalts：指定用户中密码所使用的 cleartext。（我不大清楚此功能的作用）。
- -timeout:<几分钟>：指定解密持续的时间是几分钟，到时间 JOHN 自动停止运行。
- -list：在解密过程中在屏幕上列出所有正在尝试使用的密码，建议不要使用，它会将大部分时间浪费在显示上，极大地拖慢解密速度。一般只是适用于重定向输出到文件后，检验用户所设定的某些模式是否正常。
- -beep-quiet：当解密出密码时是否要让 PC 喇叭叫一下，以提醒用户。
- -noname-nohash：不使用内存来保存“用户名”等内容。
- -des-md5：指定使用的解密方式是解 DES 还是 MD5，对于解密 DES 密码不用理会这一选项。

（2）其他实用工具。

除了口令破解程序之外，在这个软件包中，还包含了其他几个实用工具，它们对于实现口令破解都有一定的帮助，这些工具都放置在run目录下，下面分别予以简要介绍。

① unshadow PASSWORD-FILE SHADOW-FILE

unshadow命令将passwd文件和shadow文件组合在一起，其结果用于John破解程序。通常应该使用重定向方法将这个程序的结果保存在文件中，之后将文件传递给John破解程序。

② unafs DATABASE-FILE CELL-NAME

unafs从二进制AFS数据库中提取口令散列值，并生成John可用的输出，通常应该把这个输出重定向到文件中。

③ unique OUTPUT-FILE

删除字典表中的重复词汇，但不改变字典表中各词条项的顺序。

（4）密码分析及检验

安装好后，我们可以灵活使用以下几种方式对自己的账户密码进行测试。

（1）简单解密方式。

通常情况下，许多用户的密码命名方式非常简单，比如foo、hello、world等，或者很多都是与用户名相同的密码口令，那么我们一般可以先采用简单解密方式来对系统中的密码进行简单的初步试探，如果发现能够成功破解，那么就需要对这些密码口令的强度进行加强，如下所示：

```
#./john -single "/etc/shadow"
Loaded 2 password hashes with 3 different salts (FreeBSD MD5 [32/32])
liyang           (liyang)
guesses: 1  time: 0:00:00:00 100%  c/s: 6975  trying: 999991900
```

在上述命令中，我们发现系统存在一个liyang用户，其用户名和密码均为liyang，因而通过最简单的方式便能将其发现和利用，如果为黑客破解则将造成不可设想的后果，因而我们的用户应该立即根据此种情况进行口令加强。

（2）字典解密方式。

用户可以使用字典文件来对系统用户的恶密码强度进行试探和测试。人们常用hello、superman、cooler、asdfgh、123456 等作为自己的密码。而-rules参数则在此基础上再加上些变化，如字典中有单词cool，则JOHN还会尝试使用cooler、CoOl、Cool等单词变化进行解密。一般视SHADOW中的用户多少及用户的字典大小、用户的机器速度，解密时间从几小时到几天不等。下面给出使用该方式进行解密的例子，假设我们已经生成了一个password.lst文件，其中包含了常用的以字典单词为依据的密码，那么我们对系统中的用户密码使用该方式进行试探性破解，由于字典中保留了young这样一个单词，因而用户google的密码所以也被试探出来，网络管理员同样需要对该密码进行加固，比如添加适当的后缀、字母和数字等：

```
# ./john --wordlist=password.lst "/etc/shadow"
Loaded 2 password hashes with 2 different salts (FreeBSD MD5 [32/32])
young          (google)
guesses: 1  time: 0:00:00:01 100%  c/s: 3571  trying: zhongguo
```

（3）其他方法。

当然，上述两种只是非常直观和简单的方法，但是如果字典足够完整和实用的话，那么就能够查出绝大多数的脆弱口令，在实践中，我们还可以综合使用以下四个最常见的选项，来对系统密码强度进行更为充分的检查和验证。

- -rules：在解密过程中使用单词规则变化功能。如将尝试 cool 单词的其他可能，如 COOLER、Cool 等，详细规则可以在 JOHN.INI 文件中的[List.Rules:Wordlist]部分查到，我们在下面将详细解释。
- -incremental[:<模式名称>]：使用遍历模式，就是组合密码的所有可能情况，同样可以在 JOHN.INI 文件中的[Incremental:*****]部分查到，我们在下面将详细解释。
- -external:<模式名称>：使用自定义的扩展解密模式，用户可以在 john.ini 中定义自己需要的密码组合方式。JOHN 也在 INI 文件中给出了几个示例，在 INI 文件的[List.External:******]中所定义的自订破解功能。
- -restore[:<文件名>]：继续上次的破解工作，JOHN 被中断后，当前的解密进度情况被存放在 RESTORE 文件中，用户可以拷贝这个文件到一个新的文件中。如果参数后不带文件名，JOHN 默认使用 RESTORE 文件。

## 5.6 企业 Linux 日志安全管理

### 5.6.1 Linux 下的日志分类

日志主要的功能有：审计和监测。它还可以实时地监测系统状态；监测和追踪侵入者等。成功地管理任何系统的关键之一，是要知道系统中正在发生什么事。Linux中提供了异常日志，并且日志的细节是可配置的。Linux日志都以明文形式存储，所以用户不需要特殊的工具就可以搜索和阅读它们。还可以编写脚本，来扫描这些日志，并基于它们的内容去自动执行某些功能。Linux日志存储在/var/log目录中。这里有几个由系统维护的日志文件，但其他服务和程序也可能会把它们的日志放在这里。大多数日志只有root账户才可以读，不过修改文件的访问权限就可以让其他人可读。在Linux系统中，有四类主要的日志。

- 连接时间日志：由多个程序执行，把记录写入到/var/log/wtmp 和/var/run/utmp，login 等程序更新 wtmp 和 utmp 文件，使系统管理员能够跟踪谁在何时登录到系统。
- 进程统计：由系统内核执行。当一个进程终止时，为每个进程往进程统计文件（pacct 或 acct）中写一个记录。进程统计的目的是为系统中的基本服务提供命令使用统计。
- 错误日志：由 syslogd（8）守护程序执行。各种系统守护进程、用户程序和内核通过 syslogd（3）守护程序向文件/var/log/messages 报告值得注意的事件。另外有许多 UNIX 程序创建日志。像 HTTP 和 FTP 这样提供网络服务的服务器也保持详细的日志。
- 实用程序日志：许多程序通过维护日志来反映系统的安全状态。su 命令允许用户获得另一个用户的权限，所以它的安全很重要，它的文件为 sulog。同样的还有 sudolog。另外，诸如 Apache 等 HTTP 的服务器都有两个日志：access_log（客户端访问日志）以及 error_log（服务出错日志）。FTP 服务的日志记录在 xferlog 文件当中，Linux 下邮件传送服务（sendmail）的日志一般存放在 maillog 文件当中。

上述四类日志中，常用的日志文件如表 5-6 所示。

表 5-6 Linux 系统中常用的日志文件

| 日志文件 | 注　释 |
|---|---|
| access-log | 记录 HTTP/Web 的传输 |
| acct/pacct | 记录用户命令 |
| boot.log | 记录 Linux 系统开机自检过程显示的信息 |
| lastlog | 记录最近几次成功登录的事件和最后一次不成功的登录 |
| messages | 从 syslog 中记录信息（有的链接到 syslog 文件） |
| sudolog | 记录使用 sudo 发出的命令 |
| sulog | 记录使用 su 命令的使用 |
| syslog | 从 syslog 中记录信息 |
| utmp | 记录当前登录的每个用户的信息 |
| wtmp | 一个用户每次登录进入和退出时间的永久记录 |
| xferlog | 记录 FTP 会话信息 |
| maillog | 记录每一个发送到系统或从系统发出的电子邮件的活动。它可以用来查看用户使用哪个系统发送工具或把数据发送到哪个系统 |

### 5.6.2 使用基本命令进行日志管理

utmp、wtmp日志文件是多数Linux日志子系统的关键，它保存了用户登录进入和退出的记录。有关当前登录用户的信息记录在文件utmp中；登录进入和退出记录在文件wtmp中；数据交换、关机以及重启的机器信息也都记录在wtmp文件中。所有的记录都包含时间戳。时间戳对于日志来说非常重要，因为很多攻击行为分析都与时间有极大的关系。这些文件在具有大量用户的系统中增长十分迅速。例如wtmp文件可以无限增长，除非定期截取。许多系统以一天或者一周为单位把wtmp配置成循环使用。它通常由cron运行的脚本来修改，这些脚本重新命名并循环使用wtmp文件。通常，wtmp在第一天结束后命名为wtmp.1；第二天后wtmp.1 变为wtmp.2 等，用户可以根据实际情况来对这些文件进行命名和配置使用。

utmp文件被各种命令文件使用，包括who、w、users和finger。而wtmp文件被程序last和ac使用。

wtmp和utmp文件都是二进制文件，它们不能被诸如tail命令剪贴或合并（使用cat命令）。用户需要使用who、w、users、last和ac来使用这两个文件包含的信息。

1. who 命令

who命令查询utmp文件并报告当前登录的每个用户。who的默认输出包括用户名、终端类型、登录日期及远程主机。使用该命令，系统管理员可以查看当前系统存在哪些不法用户，从而对其进行审计和处理。例如：运行who命令显示如下所示：

```
# who
root     pts/1       2010-02-22 13:02 (:0.0)
root     pts/2       2010-02-22 15:57 (:0.0)
root     pts/3       2010-02-22 15:57 (:0.0)
```

如果指明了wtmp文件名，则who命令查询所有以前的记录。命令who /var/log/wtmp将报告自从wtmp文件创建或删改以来的每一次登录。例如：运行该命令如下所示：

```
# who /var/log/wtmp
root     :0         2010-01-24 21:47
root     pts/1       2010-01-24 21:47 (:0.0)
root     :0         2010-02-20 19:36
root     pts/1       2010-02-20 19:36 (:0.0)
root     :0         2010-02-21 15:21
root     pts/1       2010-02-21 15:56 (:0.0)
root     pts/2       2010-02-21 16:03 (:0.0)
root     :0         2010-02-22 13:01
root     pts/1       2010-02-22 13:02 (:0.0)
root     pts/2       2010-02-22 15:57 (:0.0)
root     pts/3       2010-02-22 15:57 (:0.0)
```

2. users 命令

users用单独的一行打印出当前登录的用户，每个显示的用户名对应一个登录会话。如果一个用户有不止一个登录会话，那他的用户名将显示相同的次数。运行该命令将如下所示：

```
# users
root root root
```

### 3. last 命令

last命令往回搜索wtmp来显示自从文件第一次创建以来登录过的用户。系统管理员可以周期性地对这些用户的登录情况进行审计和考核，从而发现其中存在的问题，确定不法用户，并进行处理。运行该命令，如下所示：

```
# last
root     pts/3        :0.0            Mon Feb 22 15:57   still logged in
root     pts/2        :0.0            Mon Feb 22 15:57   still logged in
root     pts/1        :0.0            Mon Feb 22 13:02   still logged in
root     :0                          Mon Feb 22 13:01   still logged in
reboot   system boot  2.6.18-8.el5    Mon Feb 22 12:56          (03:02)
root     pts/2        :0.0            Sun Feb 21 16:03 - down   (02:37)
root     pts/1        :0.0            Sun Feb 21 15:56 - down   (02:45)
root     :0                          Sun Feb 21 15:21 - down   (03:20)
reboot   system boot  2.6.18-8.el5    Sun Feb 21 15:19          (03:22)
root     pts/1        :0.0            Sat Feb 20 19:36 - down   (01:50)
root     :0                          Sat Feb 20 19:36 - down   (01:51)
reboot   system boot  2.6.18-8.el5    Sat Feb 20 19:34          (01:53)
root     pts/1        :0.0            Sun Jan 24 21:47 - down   (00:02)
root     :0                          Sun Jan 24 21:47 - down   (00:02)
reboot   system boot  2.6.18-8.el5    Sun Jan 24 21:45          (00:05)
reboot   system boot  2.6.18-8.el5    Sun Jan 24 21:41          (00:02)

wtmp begins Sun Jan 24 21:41:03 2010
```

读者可以看到，使用上述命令显示的信息太多，区分度很小。所以，可以通过指明用户来显示其登录信息即可。例如，使用last reboot来显示reboot的历史登录信息，则如下所示：

```
# last reboot
reboot   system boot  2.6.18-8.el5    Mon Feb 22 12:56          (03:07)
reboot   system boot  2.6.18-8.el5    Sun Feb 21 15:19          (03:22)
reboot   system boot  2.6.18-8.el5    Sat Feb 20 19:34          (01:53)
reboot   system boot  2.6.18-8.el5    Sun Jan 24 21:45          (00:05)
reboot   system boot  2.6.18-8.el5    Sun Jan 24 21:41          (00:02)

wtmp begins Sun Jan 24 21:41:03 2010
```

### 4. ac 命令

ac命令根据当前的/var/log/wtmp文件中的登录进入和退出来报告用户连接的时间（小时），如果不使用标志，则报告总的时间。例如：ac（回车）显示：total 18.47，如下所示：

```
# ac
total      18.47
```

另外，可加一些参数，例如，last -u 102 将报告UID为 102 的用户；last -t 7 表示限制上一周的报告。

5. lastlog 命令

lastlog文件在每次有用户登录时被查询。可以使用lastlog命令检查某特定用户上次登录的时间，并格式化输出上次登录日志/var/log/lastlog的内容。它根据UID排序显示登录名、端口号（tty）和上次登录时间。如果一个用户从未登录过，lastlog显示**Never logged**。注意需要以root身份运行该命令。运行该命令如下所示：

```
# lastlog
用户名            端口         来自           最后登录时间
root                                       **从未登录过**
bin                                        **从未登录过**
daemon                                     **从未登录过**
adm                                        **从未登录过**
lp                                         **从未登录过**
sync                                       **从未登录过**
shutdown                                   **从未登录过**
halt                                       **从未登录过**
mail                                       **从未登录过**
news                                       **从未登录过**
uucp                                       **从未登录过**
operator                                   **从未登录过**
games                                      **从未登录过**
gopher                                     **从未登录过**
ftp                                        **从未登录过**
nobody                                     **从未登录过**
rpm                                        **从未登录过**
dbus                                       **从未登录过**
avahi                                      **从未登录过**
mailnull                                   **从未登录过**
smmsp                                      **从未登录过**
nscd                                       **从未登录过**
vcsa                                       **从未登录过**
haldaemon                                  **从未登录过**
rpc                                        **从未登录过**
rpcuser                                    **从未登录过**
sshd                                       **从未登录过**
```

```
pcap                                  **从未登录过**
ntp                                   **从未登录过**
gdm                                   **从未登录过**
apache                                **从未登录过**
distcache                             **从未登录过**
postgres                              **从未登录过**
mysql                                 **从未登录过**
dovecot                               **从未登录过**
webalizer                             **从未登录过**
squid                                 **从未登录过**
named                                 **从未登录过**
xfs                                   **从未登录过**
sabayon                               **从未登录过**
postfix                               **从未登录过**
amanda                                **从未登录过**
cyrus                                 **从未登录过**
mailman                               **从未登录过**
radiusd                               **从未登录过**
exim                                  **从未登录过**
privoxy                               **从未登录过**
quagga                                **从未登录过**
radvd                                 **从未登录过**
ldap                                  **从未登录过**
tomcat                                **从未登录过**
pegasus                               **从未登录过**
liyang                                **从未登录过**
google                                **从未登录过**
```

### 5.6.3 使用 syslog 设备

syslog已被许多日志函数采纳，它用在许多保护措施中，任何程序都可以通过syslog记录事件。syslog可以记录系统事件，可以写到一个文件或设备中，或给用户发送一个信息。它能记录本地事件或通过网络记录另一个主机上的事件。syslog是一种工业标准协议，可用来记录设备的日志。在UNIX系统、路由器、交换机等网络设备中，系统日志（System Log）记录系统中任何时间发生的大小事件。管理者可以通过查看系统记录，随时掌握系统状况。UNIX的系统日志是通过syslogd进程记录系统有关事件，也可以记录应用程序运作事件。通过适当的配置，我们还可以实现运行syslog协议的机器间通信，通过分析这些网络行为日志，来追踪掌握与设备和网络有关的状况。

### 1. 设定 syslog 配置文件

syslog设备依据两个重要的文件：/etc/syslogd守护进程和/etc/syslog.conf配置文件。通常情况下，多数syslog信息被写到/var/adm或/var/log目录下的信息文件（messages.*）中。一个典型的syslog记录包括生成程序的名字和一个文本信息，它还包括一个设备和一个优先级范围。

通过使用syslog.conf文件，可以对生成的日志的位置及其相关信息进行灵活的配置，满足应用的需要。该配置文件指明了syslogd守护程序记录日志的行为，该程序在启动时查询配置文件。该文件由不同程序或消息分类的单个条目组成，每个占一行。对每类消息提供一个选择域和一个动作域。这些域由tab隔开。

- 选择域指明消息的类型和优先级。
- 动作域指明 syslogd 接收到一个与选择标准相匹配的消息时所执行的动作。

syslog.conf行的基本语法是：

```
消息类型.优先级 动作域
```

其中，每个选择域是由消息类型和优先级组成。当指明一个优先级时，syslogd将记录一个拥有相同或更高优先级的消息。Linux中一些主要的消息类型如表 5-7 所示，表 5-8 列出了一些优先级信息。

表 5-7　syslog 消息类型

| 消息类型 | 消息来源 |
|---|---|
| kern | 内核 |
| User | 用户程序 |
| Damon | 系统守护进程 |
| Mail | 电子邮件系统 |
| Auth | 与安全权限相关的命令 |
| Lpr | 打印机 |
| News | 新闻组信息 |
| Uucp | Uucp 程序 |
| Cron | 记录当前登录的每个用户信息 |
| wtmp | 一个用户每次登录进入和退出时间的永久记录 |
| Authpriv | 授权信息 |

表 5-8　syslog 常用优先级

| 优 先 级 | 描　述 |
|---|---|
| emerg | 最高的紧急程度状态 |
| alert | 紧急状态 |
| Cirt | 重要信息 |
| warning | 警告 |
| err | 临界状态 |
| notice | 出现不寻常的事情 |
| info | 一般性消息 |
| Debug | 调试级信息 |
| None | 不记录任何日志信息 |

不同的服务类型有不同的优先级，数值较大的优先级涵盖数值较小的优先级。如果某个选择条件只给出了一个优先级而没有使用任何优先级限定符，对应于这个优先级的消息以及所有更紧急的消息类型都将包括在内。比如说，如果某个选择条件里的优先级是"warning"，它实际上将把"warning"、"err"、"crit"、"alert"和"emerg"都包括在内。

syslog允许人们使用三种限定符对优先级进行修饰：星号（*）、等号（=）和叹号（!）。

- 星号（*）的含义是把本项服务生成的所有日志消息都发送到操作动作指定的地点。就像它在规则表达式里的作用一样，星号代表"任何东西"。在前面给出的例子里，"mail.*"将把所有优先级的消息都发送到操作动作指定的/var/log/mail 文件里。使用"*"限定符与使用"debug"优先级的效果完全一样，后者也将把所有类型的消息发送到指定的地点。
- 等号（=）的含义是只把本项服务生成的本优先级的日志消息都发送到操作动作指定的地点。比如说，可以用"="限定符只发送调试消息而不发送其他更紧急的消息（这将为应用程序减轻很多负担）。当你只需要发送特定优先级别的消息时，就要使用等号限定符。
- 叹号（!）的含义是把本项服务生成的所有日志消息都发送到操作动作指定的地点，但本优先级的消息不包括在内。

所以，根据上面介绍的相关知识，我们给出以下例子作为示范。

（1）如果指明\"crit\"，那所有标为crit、alert和emerg的消息将被记录。每行的行动域指明当选择域选择了一个给定消息后应该把其发送到哪儿。例如，如果想把所有邮件消息记录到一个文件中，如下所示：

```
#Log all the mail messages in one place
mail.*                          /var/log/maillog
```

（2）其他设备也有自己的日志。UUCP和news设备能产生许多外部消息。它把这些消息存到自己的日志（/var/log/spooler）中并把级别限为\"err\"或更高。例如：

```
# Save news errors of level crit and higher in a special file.
uucp,news.crit                     /var/log/spooler
```

（3）当一个紧急消息到来时，可能想让所有的用户都得到。也可能想让自己的日志接收并保存。

```
#Everybody gets emergency messages， plus log them on anther machine
*.emerg *
*.emerg @linuxaid.com.cn
```

（4）有时syslogd将产生大量的消息。例如内核（\"kern\"设备）可能很冗长，用户很难看得清楚明了，那么用户可能想把内核消息记录到/dev/console中。下面的例子表明内核日志记录被注释掉了：

```
#Log all kernel messages to the console
#Logging much else clutters up the screen
#kern.* /dev/console
```

（5）用户可以在一行中指明所有的设备。下面的例子把info或更高级别的消息送到/var/log/messages，除了mail以外。级别\"none\"禁止一个设备：

```
#Log anything (except mail) of level info or higher
```

```
#Don\'t log private authentication messages!
*.info:mail.none;autHPriv.none /var/log/messages
```

### 2. 使用 syslog 进程

syslogd守护程序是由/etc/rc.d/init.d/syslog脚本在运行级 2 下被调用的，默认不使用选项。但有两个选项-r和-h很有用。

- 如果将要使用一个日志服务器，必须调用 syslogd -r。默认情况下 syslogd 不接受来自远程系统的信息。当指定-r 选项，syslogd 将会监听从 514 端口上进来的 UDP 包。
- 如果还希望日志服务器能传送日志信息，可以使用-h 标志。默认时，syslogd 将忽略使其从一个远程系统传送日志信息到另一个系统的 syslogd。

另外，如果需要重新启动syslog守护进程（/etc/syslog.conf的修改只有在syslog守护进程重新启动后才会生效），并且只想重新启动syslog守护进程而不是整个系统，在Red Hat Linux机器上，执行以下两条命令之一即可：

```
/etc/rc.d/init.d/syslogstop;/etc/rc.d/init.d/syslogstart
/etc/rc.d/init.d/syslogrestart
```

### 3. 实际应用中的 syslog 调用接口

在实际的使用过程中，我们可以根据 5.6.3 节中的介绍通过配置文件和查看相应的日志文件来使用syslog。然而，在许多应用场景下，我们往往需要通过程序产生输出信息并进行记录，也就是说要把一些信息写成日志文件，正常情况下运行程序的人不用关心日志里的内容，只有在出现问题的时候才会查看日志文件里的内容以确定问题所在。因此，本节将介绍如何通过syslog日志系统提供的API调用接口，来使用程序实现对syslog的使用。

1）主要的函数

在Linux中，提供了四个有关syslog日志系统的系统调用，供用户使用。

- openlog：打开日志设备，以供读取和写入，与文件系统调用的 open 类似。
- syslog：写入日志，与文件系统调用的 write 类似。
- closelog：关闭日志设备，与文件系统调用的 close 类似。
- vsyslog：它和 syslog 功能一样，负责写入日志，只是参数格式不同。

（1）openlog函数。

openlog函数的声明如下：

```
void openlog(const char *ident, int option, int facility);
```

此函数用来打开一个到系统日志记录程序的连接，打开之后就可以用syslog或vsyslog函数向系统日志里添加信息了。而closelog函数就是用来关闭此连接的。

openlog的第一个参数ident是一个标记，ident所表示的字符串将固定地加在每行日志的前面以标识这个日志，通常就写成当前程序的名称以作标记。

第二个参数option一般是下列选项值取“与”运算（使用“|”表示，如“LOG_CONS | LOG_PID”）的结果。

- LOG_CONS：如果送到 system logger 时发生问题，直接写入系统终端。
- LOG_NDELAY：立即开启连接，通常连接是在第一次写入消息时才打开的。
- LOG_PERROR：将消息也同时送到 stderr 设备。

- LOG_PID：将进程 PID 含入所有消息中。

第三个参数facility指明记录日志的程序类型，它主要具有以下几类日志类型。

- LOG_AUTH ：安全/授权消息。
- LOG_AUTHPRIV：安全/授权消息。
- LOG_CRON：时间守护进程（cron 和 at）专用。
- LOG_DAEMON：其他系统守护进程。
- LOG_KERN：核心消息。
- LOG_LOCAL0 到 LOG_LOCAL7：系统保留。
- LOG_LPR：printer 子系统。
- LOG_MAIL：mail 子系统。
- LOG_NEWS：USENET 新闻子系统。
- LOG_SYSLOG：syslogd 进程内部所产生的消息。
- LOG_USER（默认）：一般使用者默认使用消息。
- LOG_UUCP：UUCP 子系统。
- LOG_FTP：FTP 子系统使用。

（2）syslog函数。

syslog函数的声明如下：

```
void syslog(int priority, const char * message, ...);
```

第一个参数是消息的紧急级别priority，第二个参数是消息及其格式，之后是格式对应的参数，如同C语言里面printf输出函数一样使用，具体的格式这里就不再详述，它不是本书介绍的重点。

这里还需要详细介绍一下第一个参数priority，它是由severity level和facility组成的。facility已经在上面介绍了，下面介绍一下severity level，也就是消息的重要级别，它主要包括：

- LOG_EMERG：紧急状况；
- LOG_ALERT：高优先级问题，比如说数据库崩溃等，必须要立即采取反应行动；
- LOG_CRIT：重要状况发生，比如硬件故障；
- LOG_ERR：错误发生；
- LOG_WARNING：警告发生；
- LOG_NOTICE：一般状况，需要引起注意；
- LOG_INFO：信息状况；
- LOG_DEBUG：调试消息。

在实际使用中，如果我们的程序要使用系统日志功能，只需要在程序启动时使用openlog函数来连接syslogd程序，后面随时用syslog函数写日志就行了。

（3）closelog函数。

相对上述两个函数来说，该函数非常简单，其声明如下：

```
void  closelog(  void  );
```

值得注意的是，虽然该函数的使用和调用情况非常简单，但却是必不可少的，因为在Linux系统中，打开的日志也是资源，如果只使用openlog函数打开日志，而忘记使用closelog关闭日志的话，当打开的日志数量累积到一定程度，便会造成内存不足，从而引起系统问题。所以，

提醒用户在使用中特别注意。

2）一个实际的程序调用例子

下面给出一个使用上述几个函数写日志的例子，以供大家参考：

```
#include <syslog.h>
#include <stdio.h>
#include <unistd.h>

int main(void)
{
       int log_test;



/*打开日志*/
       openlog("log_test ", LOG_PID|LOG_CONS, LOG_USER);

/*写日志*/
       syslog(LOG_INFO, "PID information, pid=%d", getpid());

       syslog(LOG_DEBUG,"debug message ");

/*关闭日志*/
       closelog();

}
```

## 5.7 应用 LIDS 进行 Linux 系统入侵检测

### 5.7.1 LIDS 简介

LIDS是Linux下的入侵检测和防护系统，是Linux内核的补丁和安全管理工具，它增强了内核的安全性，它在内核中实现了参考监听模式以及强制访问控制（Mandatory Access Control）模式。区别于本书将在后面部分介绍的Snort入侵检测系统，它属于网络IDS范畴，而LIDS则属于主机IDS范畴。

一般来说，LIDS的主要功能包括以下几方面。

- 重要系统资源保护：保护硬盘上任何类型的重要文件和目录，如/bin、/sbin、/usr/bin、/usr/sbin、/etc/rc.d 等目录和其下的文件，以及系统中的敏感文件，如 passwd 和 shadow 文件，防止未被授权者（包括 root 用户）和未被授权的程序进入。保护重要进程不被终止，任何人包括 root 也不能杀死进程，而且可以隐藏特定的进程。防止非法程序的 I/O 操作，保护硬盘，包括 MBR 保护等。

- 入侵检测：LIDS 可以监测到系统上任何违反规则的进程。
- 入侵响应：来自内核的安全警告，当有人违反规则时，LIDS 会在控制台显示警告信息，将非法的活动细节记录到受 LIDS 保护的系统 log 文件中。LIDS 还可以将 log 信息发到用户的信箱中。并且，LIDS 还可以马上关闭与用户的会话。

### 5.7.2 安装 LIDS

#### 1. 打补丁并配置 Linux 内核选项安装

LIDS通常需要下载其最新版本的LIDS内核补丁包，然后进行安装。下载的网站为：http://www.lids.org/，目前网站上的最新版本为：lids-2.2.3rc7-2.6.28.patch。首先将下载的LIDS内核补丁包保存到/usr/src目录下，然后以根用户的权限进入命令行模式按以下步骤进行操作。

（1）假设系统内核文件在/usr/src/Linux目录下，通过下列命令安装LIDS内核补丁包：

```
# cd /usr/src/Linux
# patch p1 < /usr/src/ lids-2.2.3rc7-2.6.28.patch
```

（2）编辑内核，选取相关选项：

```
//进入编辑内核界面
# make menuconfig
```

进入内核编译菜单界面后，建议把有关LIDS的所有项都选中。这样做的目的会让不太熟悉内核编译的用户省去很多不必要的麻烦，并且将所有的LIDS项都选择上也不会占用多少内核空间，对加入LIDS后的内核性能也不会产生多大影响。下面对一些选项进行解释：

```
Prompt for development and/or incomplete code/drivers
Sysctl support
Linux Intrusion Detection System support.
[ ] Hang up console when raising a security alert
当收到一个安全警告挂起控制台
[ ] Security alert when execing unprotected programs before sealing LIDS
 当执行没有受 LIDS 保护的程序时发送安全警告
[ ] Do not execute unprotected programs before sealing LIDS
在安装 LIDS 前不执行没有受保护的程序
[ ] Try not to flood logs
尽量不要让日志溢出
[ ] Allow switching LIDS protections
允许转换 LIDS 保护
[ ] Allow remote users to switch LIDS protections
允许远程用户来转换 LIDS 保护
[ ] Allow any program to switch LIDS protections
允许任何程序来转换 LIDS 保护
[ ] Allow reloading config. File
允许重新引导配置文件
[ ] Port Scanner Detector in kernel
```

```
内核的端口扫描器
[ ] Send security alerts through network
通过网络发送安全警告
[ ] Hide klids kernel thread
隐藏内核进程
[ ] Use generic mailer pseudo-script
使用通用的邮件发送脚本
```

（3）在选择好要加入到内核中的LIDS项后，就可以通过下列命令重新编译内核：

```
# make dep
# make clean
# make bzImage
# make modules
# make modules_install
```

完成上述内核编译工作后，一个加入了LIDS的内核就重新编译好了。要使加入了LIDS的新内核工作，必须重新启动系统。

#### 2. 源代码安装 LIDS 工具包

首先从上述网站上下载LIDS工具包的安装文件，目前网站上的最新版本为：lids-2.3.rc7-2.6.28.patch，然后按以下步骤安装它。

（1）解压缩源码包：

```
# tar -zxvf lids-2.3.rc7-2.6.28.patch
```

（2）切换目录并生成makefile文件：

```
# cd lids-2.3.rc7-2.6.28
# ./configure
```

（3）安装：

```
# make
# make install
```

这样就会将Lidsadm和Lidsconf这两个工具安装到/sbin/目录中，同时会创建一个/etc/lids的目录，并会在此目录下生成一个默认的配置文件。

### 5.7.3 配置和使用 LIDS

#### 1. 基本配置

必须配置LIDS系统，使其符合用户的安全需要。用户可以定义受保护的文件、受保护的进程等。

首先，更新默认lids.conf的inode/dev值。

```
# /sbin/lidsadm -U
```

然后，获得一个RipeMD-160 加密口令：

```
# /sbin/lidsadm -P
```

默认情况下，Lidsadm将把默认配置文件安装到/etc/lids/。用户必须根据自己的需要重新配

置。当内核启动时，配置信息就把相关信息读入内核来初始化LIDS系统。需要特别注意该目录中的以下几个相关的配置文件。

- lids.conf：这个文件用来存储 LIDS ACLs 信息。它包括定义对象访问类型的 ACLs（访问控制列表）。
- lids.cap：这个文件包括系统的所有性能，可以编辑这个文件来配置这些性能。
- lids.net：这个文件用来配置发给管理员信箱的警告信息。用户可以定义 SMTP 服务器、端口、消息头等。仅在配置内核时，选择了 Send security alerts through network 内核配置选项才有该文件。
- lids.pw：这个文件存储由"lidsadm -P"命令生成的密码文件。配置内核时需要选择 Allow switching LIDS protections 选项，就必须有该文件。

### 2. Lidsadm 工具

Lidsadm是LIDS的管理工具单元，可以用它来管理系统中的LIDS。Lidsadm的主要作用就是启用或停用LIDS，以及封存LIDS到内核中和查看LIDS状态。

使用下列命令可以列出Lidsadm的所有可用选项：

```
# lidsadm -h
```

其常用命令参数的具体含义如下。

- -s：开关某些保护选项时指示应提交密码。
- -I：开关某些保护选项时不提交密码。
- LIDS_FLAG：为 Lidsadm 的标志值。
- -v：显示版本。
- -V：查看现在 LIDS 状态。
- -h：列出所有选项。

Lidsadm还包括了许多常用的部分主要功能模块，它们的列表和主要功能说明如表 5-9 所示。

表 5-9　Lidsadm 主要功能模块说明

| 功能模块 | 说　明 |
|---|---|
| CAP__CHOWN | 修改目录或文件的属主和组 |
| CAP__NET__BROADCAST | 监听广播 |
| CAP__NET__ADMIN | 接口、防火墙、路由器改变 |
| CAP__IPC__LOCK | 锁定共享内存 |
| CAP__SYS__MODULE | 插入和移除内核模块 |
| CAP__HIDDEN | 隐藏进程 |
| CAP__SYS__RESOURCE | 设置资源限制 |
| CAP__KILL__PROTECTED | 杀死保护进程 |
| CAP__PROTECTED | 保护进程为单用户方式 |

Lidsadm有以下几个可用的标志值（Available flags）。

- LIDS：禁止或激活本地 LIDS。

- LIDS__CLOBAL：完全禁止或激活 LIDS。
- RELOAD__CONF：重新加载配置文件。

### 3. Lidsconf 工具

Lidsconf主要用来为LIDS配置访问控制列表（ACLs）和设置密码。输入以下命令能显示Lidsconf所有的可用选项。

# lidsconf -h

此命令执行后会返回以下命令参数。

- -A：增加一条指定的选项到已有的 ACL 中。
- -D：删除一条指定的选项。
- -E：删除所有选项。
- -U：更新 dev/inode 序号。
- -L：列出所有选项。
- -P：产生用 Ripemd-160 加密的密码。
- -V：显示版本。
- -h：显示帮助。
- -H：显示更多的帮助。
- -s [--subject]：指定一个子对象，可以为任何程序，但必须是文件。
- -o[object]：可以是文件、目录或功能（capabilities）和 socket 名称。
- -j：它有以下几个参数。
    - DENY：禁止访问。
    - READONLY：只读。
    - APPEND：增加。
    - WRITE：可写。
    - GRANT：对子对象授于能力。
    - ignore：对设置的对象忽略所有权限。
    - disable：禁止一些扩展特性。
- 其他选项。
    - -d：目标的可执行 domain。
    - -i：继承级别。
    - -t：指定从某一时段到另一时段可以进行怎样的操作。
    - -e：扩展列表。

### 4. 主要使用方法

首先，用户需要根据具体的情况来确定要保护哪些文件。一般情况下，为了保证Linux系统安全，至少需要保护系统二进制文件和系统配置文件，比如，/bin、/sbin/、/usr/、/etc/、/var/log/等。

其次，需要确定以什么方式来保护文件。LIDS提供了以下三种保护类型。

（1）拒绝任何人访问：带有DENY标志的文件和目录没有人能够看见，也不能修改。那些非常敏感的文件应该加上DENY标志。其用法如下：

```
lidsconf -A -o file_to_protected -j DENY
```

例如，可以使用如下命令来拒绝用户（包括root用户）对/etc/passwd文件的访问：

```
# lidsconf -A -o /etc/ passwd -j DENY
```

在重启或重新加载配置文件后，用户将会看到相应的操作遭到LIDS的拒绝，从而保护该文件：

```
# ls /etc/passwd
ls: /etc/passwd: No such file or directory
```

（2）配置只读文件：任何用户不能改变带有只读标记的文件。比如/etc/passwd、/bin/passwd文件一般属于此类。

其用法如下：

```
lidsconf -A -o file_to_protect -j READONLY
```

例如，我们可以保护整个/bin/目录，使之只读，如下所示：

```
# /sbin/lidsconf -A -o /bin/ -j READONLY
```

也可以保护/etc/passwd文件为只读，如下所示：

```
# /sbin/lidsconf -A -o /etc/passwd -j READONLY
```

（3）只能追加的文件：一般来说，系统日志文件应定义成此类。比如/var/log/message、/var/log/secure。这些文件只能以追加的模式打开，用户不能修改前面的部分。

其用法如下：

```
lidsconf -A -o filename_to_protect -j APPEND
```

例如，我们可以保护系统日志文件，如下所示：

```
# /sbin/lidsconf -A -o /var/log/message -j APPEND
# /sbin/lidsconf -A -o /var/log/secure -j APPEND
```

我们也可以针对具体的网络服务器日志进行保护：

```
//保护 httpd 日志文件
# /sbin/lidsconf -A -o /var/log/httpd -j APPEND
//保护 vsftpd 日志文件
# /sbin/lidsconf -A -o /var/log/vsftpd -
```

# 第 6 章

# 锦上添花：企业 Linux 操作系统 ACL 应用及安全加固

本章导读

企业 Linux 操作系统的安全及其灵活应用是保障系统安全的重要手段。上一章介绍了传统的针对目录/文件的访问控制方式，本章将介绍更为灵活的通过访问控制列表（ACL）以及 SELinux 的企业 Linux 安全加固机制。

## 6.1 安全加固必要性分析

企业级Linux的访问控制在实际应用过程中存在一些问题。例如，传统的以文件和目录为导向的访问控制使用不是非常灵活，难以指定一个目录及其文件满足一个或者多个指定组的访问控制需求。另外，传统的自主访问控制策略也难以满足当前复杂环境下系统安全的防护需求。为了解决这些问题，我们提出采用以下两种方法来进行。

- 结合采用 ACL 来灵活地扩展传统的目录/文件访问控制方式，以满足企业多方面的需求。
- 使用 SELinux 强制访问控制机制来替代现有的自主访问控制方式，细粒度地实现系统安全加固。

## 6.2 加固第一步：使用 ACL 进行灵活访问控制

文件/目录访问控制是Linux操作系统安全的重要组成部分。传统的Linux操作系统支持用户－用户组－其他用户的访问控制机制，来限定系统用户对文件/目录的访问权限，该机制已经广泛为用户所接受和应用。而在实际的使用过程中，用户会意识到在很多应用场景该机制并不能灵活、高效地满足访问控制需求，因而自Linux内核 2.6 版本开始便支持更为灵活的ACL（访问控制列表）机制。本节将通过实例来详细介绍这种机制的原理及使用。

### 6.2.1 传统的用户–用户组–其他用户（U–G–O）访问控制机制回顾

#### 1. U-G-O 模式原理

在第 5 章中，我们对Linux系统的文件系统访问控制作了详细的介绍。Linux系统中的每个文件和目录都有访问许可权限，通过其确定谁可以通过何种方式对文件和目录进行访问和操作。文件或目录的访问权限分为只读、只写和可执行三种。以文件为例，只读权限表示只允许读取内容，而禁止对其做任何的更改操作；只写权限允许对文件进行任何的修改操作；可执行权限表示允许将该文件作为一个程序执行。文件被创建时，文件所有者自动拥有对该文件的读、写和可执行权限，以便于对文件的阅读和修改。用户也可根据需要把访问权限设置为需要的任何组合。

有三种不同类型的用户可对文件或目录进行访问：文件所有者、同组用户和其他用户。所有者一般是文件的创建者，它可以允许同组用户有权访问文件，还可以将文件的访问权限赋予系统中的其他用户。在这种情况下，系统中的每一位用户都能访问该用户拥有的文件或目录。

每一个文件或目录的访问权限都有三组，每组用三位表示，分别为文件属主的读、写和执行权限；与属主同组的用户的读、写和执行权限；系统中其他用户的读、写和执行权限。当用ls-l命令显示文件或目录的详细信息时，最左边的一列为文件的访问权限。例如：

```
# ls -l
总计 76
-rw------- 1 root root   797 11-06 20:41 anaconda-ks.cfg
drwxr-xr-x 2 root root  4096 11-06 13:50 Desktop
-rw-r--r-- 1 root root 44843 11-06 20:40 install.log
-rw-r--r-- 1 root root  7513 11-06 20:35 install.log.syslog
```

横线代表空许可（即表示不具有该权限）。r代表只读，w代表写，x代表可执行。注意，这里共有10个位置，第1个字符指定了文件类型。在通常意义上，一个目录也是一个文件。如果第1个字符是横线，表示是一个非目录的文件。如果是d，表示是一个目录。后面的9个字符每三个构成一组，依次表示文件主、组用户、其他用户对该文件的访问权限。

例如：

```
-rw-r--r--   install.log
```

表示文件install.log的访问权限，说明install.log是一个普通文件； install.log的属主有读写权限；与install.log属主同组的用户只有读权限；其他用户也只有读权限。

确定了一个文件的访问权限后，用户可以利用Linux系统提供的chmod命令来重新设定不同的访问权限，也可以利用chown命令来更改某个文件或目录的所有者。

### 2. 字母文件权限设定法

字母设定法的一般使用形式为：chmod [who] [+|-|=] [mode] 文件名。

其中，操作对象who可以是下述字母中的任意一个或者为各字母的组合。

- u 表示“用户（user）”，即文件或目录的所有者。
- g 表示“同组（group）用户”，即与文件属主有相同组 ID 的所有用户。
- o 表示“其他（others）用户”。
- a 表示“所有（all）用户”。其为系统默认值。

操作符号如下。

- + 添加某个权限。
- – 取消某个权限。
- = 赋予给定权限并取消其他所有权限（如果有的话）。

设置mode所表示的权限可用下述字母的任意组合。

- r：可读。
- w：可写。
- x：可执行。只有目标文件对某些用户是可执行的或该目标文件是目录时才追加 x 属性。
- s：在文件执行时把进程的属主或组 ID 置为该文件的文件属主。方式“u+s”设置文件的用户 ID 位，“g+s”设置组 ID 位。
- t：将程序的文本保存到交换设备上。
- u：与文件属主拥有一样的权限。
- g：与文件属主同组的用户拥有一样的权限。
- o：与其他用户拥有一样的权限。

### 3. 数字文件权限设定法

数字设定法是与文字设定法功能等价的设定方法，只不过比文字设定法更加简便。数字表示的属性的含义为：0 表示没有权限，1 表示可执行权限，2 表示可写权限，4 表示可读权限，然后将其相加。所以数字属性的格式应为 3 个从 0～7 的八进制数，其顺序是（u）、（g）、（o）。其他的与文字设定法基本一致。

如果想让某个文件的属主有“读/写”两种权限，需要把 4（可读）+2（可写）=6（读/写）。

数字设定法的一般形式为：chmod [mode] 文件名。

下面给出一些使用该数字设定法的例子。

（1）设定文件install.txt的属性为：文件属主（u）拥有读、写权限；与文件属主同组人用户（g）拥有读权限；其他人（o）拥有读权限。

```
#chmod 644 install.txt
```

（2）设定example.c文件的属性为：文件主本人（u）具有可读、可写、可执行权限；与文件主同组人（g）具有可读、可执行权；其他人（o）没有任何权限。

```
#chmod 750 example.c
```

### 6.2.2 扩展的访问控制列表（ACL）方式

#### 1. 为什么要采用 ACL

U-G-O访问控制机制在很多情况下难以满足实际文件/目录访问授权的需求，比如，要设定一个组中的部分用户对特定的文件/目录具有读取和访问权限（rw-），而另外一部分用户只能具备读权限（r--），这在传统的Linux访问控制中无法通过单纯地建立新的组和用户来实现。因此，为了解决这些问题，人们提出了一种新的访问控制方法，也就是访问控制列表（Access Control List，ACL）。

ACL是一个POSIX（可移植操作系统接口，Portable Operating System Interface）标准。目前，支持ACL需要内核和文件系统的支持。现在 2.6 内核配合EXT2/EXT3、JFS、XFS、ReiserFS等文件系统都是可以支持ACL的。在目前主流的发行套件，如Red Hat Enterprise Linux （RHEL）5、RHEL 6、Fedora 16 等都已经支持ACL。

#### 2. ACL 的类型及权限位

ACL（Access Control Lists）是由一系列的Access Entry所组成的。每一条Access Entry定义了特定的类别可以对文件拥有的操作权限。Access Entry主要包括六个，可分为两大类：一类包括owner、owning group和other，对应传统U-G-O机制中的user、group和other；另一类则包括named user、named group和mask。Access Entry的主要说明如下。

- user：相当于 Linux 里文件所有者的 permission。
- named user：定义了额外的用户可以对此文件拥有的 permission。
- group：相当于 Linux 里 group 的 permission。
- named group：定义了额外的组可以对此文件拥有的 permission。
- mask：定义了 named user、named group 和 group 的最大权限。
- other：相当于 Linux 里 other 的 permission。

下面举个简单的例子。对于如下的ACL Entry的定义：

```
# file: example.xml
# owner: liyang
# group: operation
user::rwx
user:shengping:rw-
group::rw-
group:dev:r-x
```

```
mask::rwx
other::r--
```

其中，前面三行以#开头的定义了文件名、文件的所有者和所有者所在的组。后面紧跟着的六行则说明了如下的问题。

- user::rwx 说明文件所有者拥有读写和执行权限。
- user:shengping:rw-定义了用户 shengping 拥有对文件的读写权限。
- group::rw-说明文件的 group 拥有读写权限。
- group:dev:r-x 定义 dev 组拥有对文件的读和执行权限。
- mask::rwx 定义了 mask 的权限为读。
- other::r-- 定义了 other 用户的权限为读。

值得特别注意的是：mask的rwx权限定义，决定了user:shengping、group和group:dev对文件的最大权限分别是rw、rw和rx。这也是mask这个ACL Entry的用途所在，可以用它来批量控制权限。当然，在实际的使用过程中，并不需要用户对该Entry直接进行操作，而只需要使用相应的setfacl命令即可，系统将会根据该命令的执行来自动生成上述Entry项，这其中就包含了mask这个Entry项。

### 3. ACL 的基本命令

ACL的主要命令有两个：getfacl和setfacl。getfacl用于获取文件获者目录的ACL权限信息，而setfacl则用于设置文件或者目录的ACL权限信息。下面分别对它们的使用进行简单介绍。

（1）getfacl：获取ACL权限信息。

getfacl命令用于获取文件的ACL权限信息，其基本用法为：getfacl [文件/目录名]。

下面举个例子。首先，使用touch命令建立一个测试文件acl_test：

```
$ touch file acl_test
```

然后，使用ls命令来查看该文件的权限访问属性，发现其并没有加入ACL权限，因为没有出现“+”符号：

```
$ ls -l acl_test
-rw-rw-r-- 1 gavin gavin 0 12-20 11:49 acl_test
```

接着，使用getfacl命令来获取文件的ACL权限信息，得到如下结果：

```
$ getfacl acl_test
# file: acl_test
# owner: gavin
# group: gavin
user::rw-
group::rw-
other::r—
```

值得注意的是：即使该文件系统上没有开启ACL选项，getfacl命令仍然可用，不过只显示默认的文件访问权限，即与ls -l显示的内容相似。

（2）setfacl：设置ACL权限。

为了设置文件的ACL权限，需要使用setfacl命令来详细设置文件的访问权限，其基本用法

如下：

```
setfacl -[参数] [文件/目录]
```

其常用的参数及作用如下。

- -m：建立一个 ACL 规则。
- -x：删除一个 ACL 规则。
- -b：删除全部的 ACL 规则。
- -set：覆盖 ACL 规则。

下面来详细介绍如何使用setfacl来设置文件/目录的ACL权限。

（3）添加/修改ACL规则。

需要使用-m选项来进行操作。

举个例子，使用该命令为用户gavin和组test设置acl_test文件的读写权限，并使用getfacl查看设置结果：

```
$ setfacl -m u:gavin:rw,g:test:r acl_test
$ getfacl acl_test
# file: acl_test
# owner: gavin
# group: gavin
user::rw-
user:gavin:rw-
group::rw-
group:test:r--
mask::rw-
other::r--
```

在上面的命令示例中，可以清楚地看到加粗部分user:gavin、group:test、mask这 3 个ACL Entry的出现，表明对文件进行了ACL权限设置，否则，不会出现该标识。为了进一步验证，我们使用ls-l来查看该文件的权限位中是否多了“+”这个标识位，如下所示：

```
$ ls -l acl_test
-rw-rw-r--+ 1 gavin gavin 0 12-20 11:49 acl_test
```

其中，user:gavin、group:test为我们设置的访问权限，而mask::rw为自动添加的内容。

（4）删除ACL规则。

使用-x选项可以方便地删除指定用户对指定文件/目录的访问权限。

以下示例删除用户gavin对文件acl_test的访问权限：

```
$ setfacl -x u:gavin acl_test
$ getfacl acl_test
# file: acl_test
# owner: gavin
# group: gavin
user::rw-
group::rw-
```

```
group:test:r--
mask::rw-
other::r--

$ ls -l acl_test
-rw-rw-r--+ 1 gavin gavin 0 12-20 11:49 acl_test
```

通过上述两段ACL权限显示的对比可以清楚地看到：用户gavin对于文件acl_test的访问权限已经完全删除了，表现为user:gavin:rw-已经不存在了。这里提醒注意的是：我们不能够通过setfacl命令来指定删除用户/组对文件/目录的某一个特定权限（如r、w或者x）。同时，也可以看到，使用ls-l命令显示文件的 9 个权限位还是没有改变，因为改变的只是ACL权限，而不是最基本的user、group和others权限。

（5）删除文件/目录的所有ACL规则。

使用-b选项可以删除文件/目录的ACL权限。以下命令将删除文件acl_test的所有ACL权限。可以看到，使用getfacl命令来查看时，mask项已经消失，即该文件已经没有了所有的ACL权限：

```
$ setfacl -b acl_test
$ getfacl acl_test
# file: acl_test
# owner: gavin
# group: gavin
user::rw-
group::rw-
other::r—
```

（6）覆盖文件的原有ACL规则。

覆盖文件的原有ACL规则需要使用--set选项。此处需要强调一下-m选项和--set选项的区别：-m选项只是修改已有的配置或是新增加一些；而--set选项和-m不同，它会把原有的ACL项全都删除，并用新的替代。另外，--set选项的参数中一定要包含U-G-O的设置，不能像-m一样只是添加ACL就可以了。

以下示例是该选项的使用方法：

```
$ setfacl --set u::rw,g::rw,o::r,u:gavin:rwx,g:test:rx acl_test
$ getfacl acl_test
# file: acl_test
# owner: gavin
# group: gavin
user::rw-
user:gavin:rwx
group::rw-
group:test:r-x
mask::rwx
other::r--
```

```
$ ls -l acl_test
-rw-rwxr--+ 1 gavin gavin 0 12-20 11:49 acl_test
```

这里需要提醒注意的是：上述acl_test文件的权限标识位中的group的rwx权限，并不是表明acl_test文件所属用户的用户组对其有x权限，实际上只是具有rw权限，而是因为在group:test:r-x中指定了test这个组具有x权限，所以ACL机制在这个标识位上进行了体现，在实际的应用中要特别注意，切记不要弄混淆了。

（7）其他选项。

除了上述介绍的 4 类用法外，setfacl还可以使用如表 6-1 所示的一些选项，供大家在实际使用中参考。

表 6-1　setfacl 选项

| 选　项 | 说　明 |
| --- | --- |
| -M | 从文件中获取待修改的 ACL 权限 |
| -X | 从文件中获取待删除的 ACL 权限 |
| -d | 设置默认的 ACL 权限 |
| -k | 删除默认的 ACL 权限 |
| -R | 递归地设置 ACL 权限 |
| --restore | 从文件恢复 ACL 权限 |

### 4. 为目录创建默认 ACL

在日常的使用过程中，经常是通过对目录来设定ACL权限来满足应用的需求，而很少仅仅通过设置特定的文件来实现，因为这样做比较繁琐和低效。因此，下面就介绍如何来为目录创建默认的ACL。

如果希望在一个目录中新建的文件和目录都使用同一个预定的ACL，那么我们可以使用默认ACL（Default ACL）。在对一个目录设置了默认的ACL以后，在目录中创建的每个文件都会自动继承目录的默认ACL作为自己的ACL。

具体的设置命令为：setfacl -d [目录名]。

下面的例子对新建的test目录进行ACL权限设置，并在其中新建了test1 和test2 文件。来看看默认ACL的设置情况。

首先，对文件夹test进行ACL权限查看：

```
$ getfacl test
# file: test
# owner: gavin
# group: gavin
user::rwx
group::rwx
other::r-x
```

接着，对其进行权限设置。

```
$ setfacl -d -m g:test:r test
$ getfacl test
```

```
# file: test
# owner: gavin
# group: gavin
user::rwx
group::rwx
other::r-x
default:user::rwx
default:group::rwx
default:group:test:r--
default:mask::rwx
default:other::r-x
```

可以看到，经过setfacl设置后，该文件夹的权限增加了以default开始的几项，表明设置成功。

然后，建立两个新的文件。查看它们的ACL权限信息：

```
$ touch test1 test2
$ getfacl test1
# file: test1
# owner: gavin
# group: gavin
user::rw-
group::rwx                    #effective:rw-
group:test:r--
mask::rw-
other::r--

$ getfacl test2
# file: test2
# owner: gavin
# group: gavin
user::rw-
group::rwx                    #effective:rw-
group:test:r--
mask::rw-
other::r--
```

可以清楚地看到：文件test1 和test2 自动继承了test设置的ACL。

#### 5. 备份和恢复 ACL

目前，Linux系统中的主要文件操作命令，如cp、mv、ls等都支持ACL。因此，在基本的文件/目录操作中可以很好地保持文件/目录的ACL权限。然而，有一些其他的命令，比如tar（文件

归档）等常见的备份工具是不会保留目录和文件的ACL信息的。因此，如果希望备份和恢复带有ACL的文件和目录，那么可以先把ACL备份到一个文件里，待操作完成以后，则可以使用--restore选项来恢复这个文件/目录中保存的ACL信息。

下面给出一个具体的例子来进行说明。

（1）获取文件acl_test的ACL权限：

```
$ getfacl acl_test
# file: acl_test
# owner: gavin
# group: gavin
user::rwx
user:test:r--
group::---
mask::r--
other::---
```

（2）将该文件的ACL权限保存至acl_test.acl文件中并进行查看：

```
$ getfacl acl_test > acl_test.acl
$ cat acl_test.acl
# file: acl_test
# owner: gavin
# group: gavin
user::rwx
user:test:r--
group::---
mask::r--
other::---
```

（3）使用tar命令进行文件操作，并查看生成文件acl.tar的ACL权限，发现其并不具备ACL权限。

```
$ tar czvf acl.tar acl_test
acl_test
$ getfacl acl.tar
# file: acl.tar
# owner: gavin
# group: gavin
user::rw-
group::rw-
other::r--
```

（4）删除acl_test文件，并释放acl.tar，查看其ACL权限，发现经过tar命令后，acl_test文件不再具备第（1）步显示的任何ACL权限，表明tar命令不能保持文件的ACL权限。

```
$ rm -rf acl_test
$ tar -xzvf acl.tar
acl_test
$ getfacl acl_test
# file: acl_test
# owner: gavin
# group: gavin
user::rwx
group::r--
other::---
```

（5）使用命令从文件恢复acl_test的ACL权限，进行查看，表明成功恢复。

```
$ setfacl --restore acl_test.acl
$ getfacl acl_test
# file: acl_test
# owner: gavin
# group: gavin
user::rwx
user:test:r--
group::---
mask::r--
other::---
```

## 6.3 加固第二步：使用 SELinux 强制访问控制

### 6.3.1 安全模型

#### 1. BLP（Bell & LaPadula）安全模型

Bell & LaPadula（BLP）模型是由David Bell和Leonard La Padula于 1973 年提出的安全模型，它同时也是一个状态机模型。它是定义多级安全性的基础，被视作基本安全公理。该公理最早是在Multics安全操作系统中得到实施。虽然从商业角度来看，该操作系统并不成功，但是单从安全性角度来看，Multics迈出了安全操作系统设计的第一步，为后来的安全操作系统的研制工作积累了大量丰富的经验。随后它又在Secure Xenix、System V/MLS、Tunis、VAX的VMM安全核心等多种安全系统的研制中得到了广泛的应用。

该模型将计算机信息系统中的实体分为两部分：主体和客体。凡是实施操作的称为主体，比如用户和进程；而被操作的对象则称为客体，比如文件、数据库等。对主体和客体而言，存在着两种最重要的安全控制方法，它们是强制存取控制以及自主存取控制方式。

- MAC（Mandatory Access Control，强制访问控制）。MAC 机制是指系统不再允许对象（比如程序、文件或文件夹等）的拥有者随意修改或授予此对象相应的权限，而是通过

强制的方式为每个对象统一授予权限，例如 SELinux 中采用的机制。该机制主要是通过“安全级”来进行。安全级包含“密级”和“部门级”两方面，密级又分为无密、秘密、机密和绝密四级。为了实施强制型安全控制，主体和客体均被赋予“安全级”。两者的关系包含三点：①主体的安全级高于客体，当且仅当主体的密级高于客体的密级，且主体的部门级包含客体的部门级；②主体可以读客体，当且仅当主体安全级高于或者等于客体；③主体可以写客体，当且仅当主体安全级低于或者等于客体。

- DAC（Discretionary Access Control，自主访问控制）。DAC 机制就是指对象（比如程序、文件或进程等）的拥有者可以任意地修改或授予此对象相应的权限。按照主体和客体的方式来说，就是主体对其拥有的客体，有权决定自己和他人对该客体应具有怎样的访问权限。例如，传统 Linux、Windows 等都采用这种机制。

图 6-1 给出了这两个概念的比较图示：

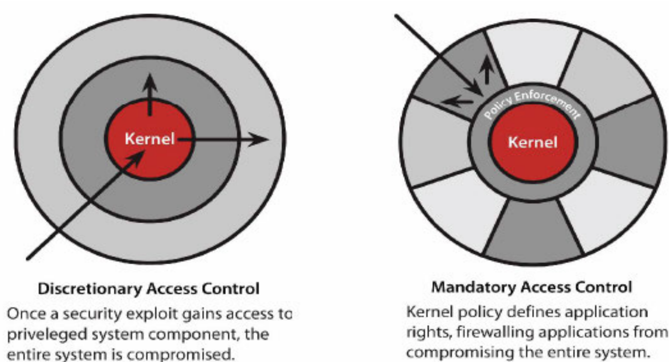



图 6-1　DAC 与 MAC 的不同之处图示

在BLP模型中，每个主体有最大安全级和当前安全级，每个客体有一个安全级。主体对客体有四种存取方式：只读、只写、执行以及读写。该模型当中有两条很重要的规则，如图 6-2 所示。

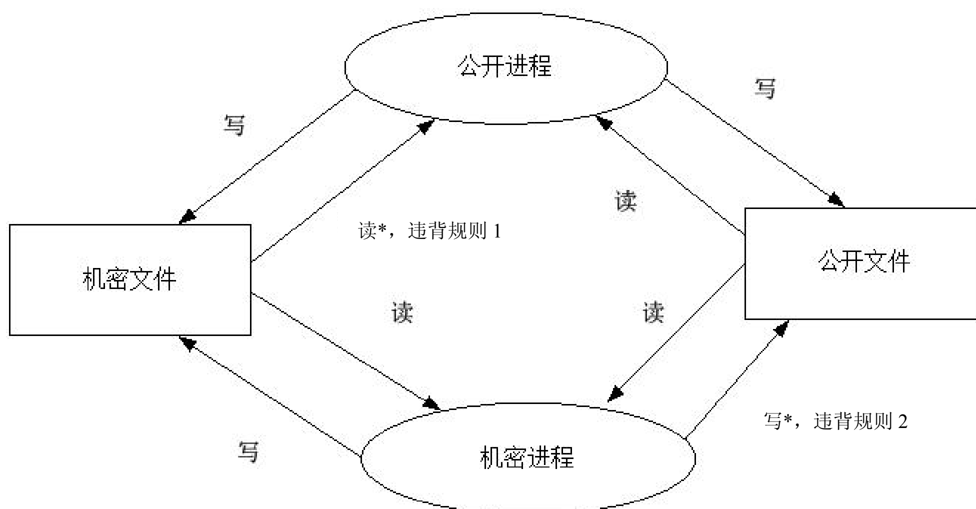



图 6-2　BLP 模型安全规则示意图

（1）简单安全特性：是指主体只能够从下读、向上写。为了使主体对客体既能读又能写，二者的安全级别必须完全一样。

（2）*——特性：主体对客体有“只写”权限，则客体安全级至少和主体的当前安全级一样高；主体对客体有“读”权限，则客体安全级应该小于或者等于主体当前安全级；主体对客体有“读写”权限，则客体安全级等于主体的当前安全级。

### 2. RBAC（Role Based Access Control）模型

基于角色的存取控制模型（RBAC）是一种强制存取控制机制。在一个RBAC系统中，根据具体的业务要求或管理要求在系统内设置若干个称之为“角色”的客体。所谓角色，用一般业务系统中的术语来说，实际上就是业务系统中的岗位、职位或者分工。例如，在一个公司内，仓库管理员、经理、会计、出纳等每一种岗位都可以设置多个职员具体从事该岗位的工作，因此他们都可以视为角色。系统管理员则负责管理系统的存取权限，将这些权限（分为不同类别和级别）分别赋予承担不同工作职责的终端用户，并可以随时根据业务的要求和变化对角色的存取权限进行管理，包括对可传递性的限制。在RBAC系统中，要求明确区分权限（Authority）和职责（Responsibility）。例如，在具有严格保密级别的系统内，访问权限为 0 级的某个官员，就不能访问保密级为 0 的所有资源，0 级是他的权限，而不是他的职责。再比如，一个用户或操作员可能有权访问某个资源的集合，但是不能涉及有关授权分配等工作；而一位主管安全的负责人可以修改访问权限，可以分配授权给各个操作员，但是不能同时具备访问/存取任何数据资源的权限。这就是角色的区分。

在RBAC系统中，将若干特定的用户集合和某种授权联结在一起，即与某种业务分工，例如将岗位、工种等相关联的授权联结在一起。这样的授权管理比针对个体的授权来说，可操作性和可管理性都强很多。角色的变动远远小于个体的变动，因此RBAC的一个主要优点就是它的简单性。美国国家标准技术研究所（NIST）对 28 个组织进行调查，结果表明，RBAC的功能相当强大，适用于许多类型（从政府机构到商业应用）的用户要求。流行的操作系统中，除Linux之外，Netware、Windows NT等也都采用了类似的RBAC技术。

在许多存取控制型系统中，以用户组作为存取控制的单位。用户组和角色最主要的区别在于，用户组作为用户的一个集合对待，并不涉及它的授权许可；而角色则既是一个用户集合，又是一个授权许可的集合。在Linux中，用户组在两个文件中加以定义（参见本书第 5 章对文件/etc/passwd和/etc/group的介绍），因此很容易判定一个用户属于哪个组，授权则以组为单位进行，系统中的每个文件或目录都与这些组确立了存取访问的授权关系，因此系统对每个用户都可以判定其是否可以访问某个文件或目录。由于授权是可以传递的，要判定某一特定的对象的授权情况需要遍历整个文件树。相对而言，判定一个用户组的成员从属关系要简单得多，对一个用户组的授权的分派指定相当不集中，尽管如此，Linux在特定情况下还可以支持角色模型的。

与一般的基于用户组的系统相比，RBAC系统要求：

（1）操作/管理角色的授权许可所花费的时空代价应当与操作/管理角色的从属关系的代价大致相当；

（2）对这两方面的操作和管理工作应该能够相对集中。

与基于安全级别和类别纵向划分的安全控制机制相比，RBAC显示了较多机动灵活性的优点，特别显著的特点是，RBAC在不同的系统配置下可以具备不同的安全控制功能，既可以构造

具备自主存取控制类型的系统，也可以构造强制存取控制类型的系统，甚至可以构造同时兼备这两种存取控制类型的系统。

#### 3. 多级别安全机制（MLS）

MLS（Multi-Level Security，多级别安全）机制给用户提供了可以用不同等级的安全水平来访问系统。例如：MLS安全分级从低到高为Confidential、Secret、Top Secret和Individuals，不同级别可查看不同的分类信息，低级别不可查看高级别的文档。

下面将要详细介绍的SELinux支持分级保护数据，它使用BLP模型，这个模型定义了系统内的信息是如何基于粘附在每个主体和客体的标签来进行流动的。如：在Secret级别的用户可与其他同级别用户共享数据，并能提取来自Confidential级（比Secret级低）的信息。但在Secret级别的进程不能查看Top Secret级别（比Secret级高）的数据。它还阻止高级别进程随便给低级别的数据写入信息。即“不能读较高级别的数据，不能写较低级别的数据”模型。

在多级别系统中，高级别用户不能自动获得管理者权限，但它们对计算机上的所有信息可以有访问权。安全上下文中，主体和客体用安全级别（Security Levels，SL）标识，SL由敏感属性和分类属性组成。Sensitivity（敏感属性）是安全体系的属性，它将数据等分成不同安全级别，如：“Secret”或“Top Secret”。Categories（分类属性）是一套非体系的属性，如：“US Only”或“UFO”，表示仅US使用。分类属性将数据分隔成几个独立的小组，每个小组可归属于不同的用户等。一个SL必须有一个敏感属性和 0 个或多个分类属性。例如：SL是{ Secret / UFO, Crypto }、{ Top Secret / UFO, Crypto, Stargate } 或{ Unclassified }。体系的敏感属性跟踪着 0 个或多个分类属性，因为敏感属性还可以分隔成彼此独立的分类属性。

客体上的安全级别称为分级（Classifications），主体上的安全级别称为Clearances。值得注意的是：MLS安全上下文至少必须有一个安全级别（它由单个敏感度和 0 个或多个范畴组成），但可以包括两个安全级别，这两个安全级别分别被叫做低（或进程趋势）和高（或进程间隙），如果高安全级别丢失，它会被认为与低安全级别的值是相同的（最常见的情况），实际上，对于客体和进程而言，低和高安全级别通常都是相同的，通常用于进程的级别范围被认为是受信任的主体（即进程信任降级信息）或多层客体，如一个目录，它又包括了不同安全级别的客体。

#### 4. 常见的操作系统加固技术

我们都知道，操作系统作为计算机系统的最为基础和重要的基础软件和系统软件，起着管理计算机资源，直接利用计算机硬件为用户提供与硬件相关、与应用无关的使用和编程接口的作用。它是上层应用软件获得高可靠性以及信息的完整性、保密性的基础。没有操作系统安全的支持，就没有获得网络安全的可能。为了计算机网络和信息的安全，我们有必要对现有操作系统（尤其是开放源码的如Linux等UNIX操作系统）的安全性进行研究和加固。

从总体上看，设计安全内核与设计操作系统类似，要用到常规的操作系统的的设计概念，但是，在设计安全内核时，优先考虑的是完整性、隔离性、可验证性等三条基本原则，而不是那些通常对操作系统来说更为重要的因素，比如灵活性、性能、开发费用、方便性等。

总体来说，对操作系统进行安全加固有以下几种方法（见图 6-3）。

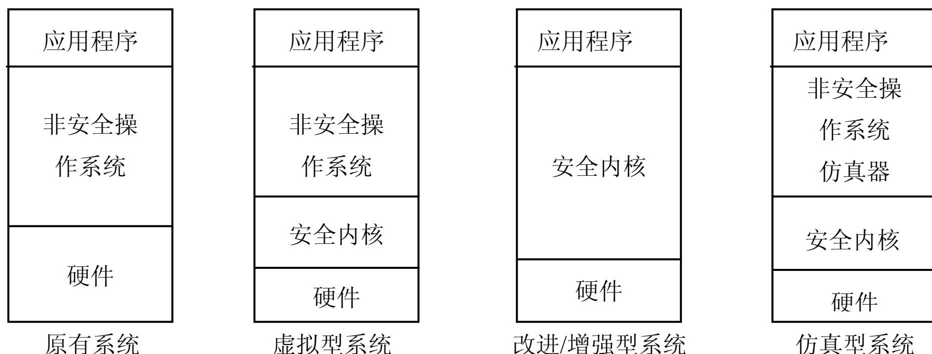



图 6-3　操作系统的几种加固方式

（1）虚拟机法。在现有的操作系统与硬件之间增加一个新的层次，作为安全内核，现有操作系统几乎可以不作任何变动而成为虚拟机。安全内核的接口基本与原有硬件等价，操作系统本身透明地在安全内核的控制之下，它可以不变地支持现有的应用程序，并且能够很好地兼容非安全操作系统的将来版本。硬件特性对虚拟机的实现非常关键，它要求原系统的硬件和结构都必须要支持虚拟机。因而这种加固方式的局限性比较大。

（2）增强法。在现有的操作系统基础上，对原有的内核以及应用程序采用一定的安全策略进行改进，并且加入相应的安全机制，形成新的操作系统内核。这样增强后的操作系统保持了原来操作系统的用户接口。这种方法是在已有操作系统基础上进行的，不是对原有内核的完全改变，因而受到原有体系结构的限制，难以达到很高的安全级别。但是这种方法并不会破坏原有系统的体系结构，开发代价小，不会影响原系统的运行效率。现在普遍采用的是这种方式。

（3）仿真法。对现有的操作系统的内核作面向安全策略的修改，然后在安全内核与原来安全操作系统的用户接口中嵌入仿真程序。那么，我们在建立安全内核的时候，可以不必如第二种方法一样受到现有应用程序的限制，而且可以完全自由地定义仿真程序于安全内核之间的接口。然而，采用这种方式需要同时设计仿真程序以及安全内核，还要受到原操作系统接口的限制，开发难度大，接口复杂。

### 6.3.2　SELinux：Linux 安全增强机制原理

随着网络的不断发展和Linux的广泛普及，Linux安全问题日益成为一个普遍关注的重要话题。传统的Linux由于root权限的权力过大而具有极大的安全威胁，一旦黑客入侵Linux操作系统并获取root权限，整个系统将暴露于恶意攻击的威胁之下。为了解决这个问题，我们可以采用SELinux（Security Enhanced Linux）。SELinux控制了无限的root权限，并不采用大众所知的传统Linux安全机制。在SELinux下的root账号采用强制访问控制机制，同时限制用户程序和系统服务器使用最低的权限做足以完成任务的工作。因此，即便系统不幸遭受黑客攻击，由于此引起的危害也随之降到最低限度，因此极大地提升了Linux的安全性能。SELinux计划早期由National Security Agency（美国国家安全局，简称为NSA）发起，旨在为Linux操作系统添加多种安全等级并更加广泛地采用于各个高安全度需求行业。早在 2005 年推出的Red Hat企业级服务器版（Red Hat Enterprise Linux，简称RHEL）Linux 4.0 就已经开始支持这种高安全性Linux，并且，Linux的

创始人Linus Torvalds表示，SELinux也已经成为Linux 2.6 系列内核的一部分。因此，在很多主流的Linux操作系统版本（如Red Hat Linux、Fedora core/Fedora、Debian等）中，都对SELinux机制进行支持。其中，在新版本的Ubuntu 6.0.4 发行版中，将包含SELinux技术。Fedora 10 中也默认安装了SELinux。

### 1. SELinux 的来源

NSA一直非常关注计算机操作系统的安全领域，他们发现大部分操作系统的安全机制，包括Windows和大部分UNIX以及Linux系统，都是以DAC机制为安全认证基础的。由于DAC机制的设计很不利于系统安全，NSA便一直致力于开发出一套好的MAC安全认证机制。

起初，NSA的MAC系统是在LOCK系统上基于类型增强而且策略灵活的强制访问控制体系，并且随后发展为两个基于Mach的系统：DTMach和DTOS，后来在犹他大学的微内核操作系统Fluke上实现出来，并命名为Flask安全体系。NSA希望有操作系统商能支持其开发的MAC机制，但由于所有的这些工作都是基于实验室，极少可以在实验室之外来进行尝试以查看这些思想在真实的应用程序的可行性，加之为保密项目，因此很难说服某个系统商采用。于是，NSA决定选择开源代码的Linux系统为实际市场试验系统。NSA把Flask安全体系集成到Linux操作系统中，SELinux便孕育而生，为了真正配合SELinux在Linux系统上的应用实施，NSA将SELinux作为一个开放源码项目发布出来，从而能获得更加广泛的开发者和用户支持。

### 2. SELinux 机制介绍

在SELinux中，每个对象（程序、文件、进程等）都拥有一个安全上下文（Security Context），它依附于每个对象身上，上面记载着这个对象所具有的权限。而我们可以通过制定安全策略（Security Policy）来定义这些安全上下文，从而定义哪种对象具有哪些权限。当一个对象需要执行某个动作时，系统会依照安全上下文所制定的内容来检查相对应的权限，如果全部权限都符合的话，系统就会允许这个操作的执行，否则都将遭到拒绝或者操作失败。这些过程不会影响到其他正常运行的对象，系统会保证它们的安全系统结构以及稳定运行。

值得注意的是：到目前为止，采用SELinux机制的Linux系统中的某个对象需要执行某个动作的时候，系统权限认证不仅仅单独根据安全上下文所制定的内容来检查，同时还要根据传统DAC来检测。只有DAC以及SELinux全部认证通过了才能进行正常操作。

如果读者了解chroot，那么就能很好地理解SELinux机制。chroot机制会把某程序置于文件系统下的某一处，当运行的时候，就不会也不能离开这个文件系统的部分，从而不会影响整个的文件系统及其权限管理。这个过程就好像是把所有的对象都分配到不同的权限自治域中，并且不同的权限自治域彼此独立，没有经过授权的对象就不允许在各个自治域之间传递数据和相关信息。

SELinux的一个重要概念是TE（Type Enforcement，类型强制），其规则是将权限与程序的访问结合在一起，而不是结合用户。本章我们讨论的所有SELinux策略语言特性都是处理主体（正常的运行中的进程）对客体（文件、目录和套接字等）的访问权的，主要集中于程序访问控制决策，这也是SELinux的主要益处，它允许SELinux策略编写者基于程序的功能和安全属性，加上用户要完成任务需要的所有访问权做出访问决策，可以将程序限制到功能合适、权限最小化的程度，因此，即使它出了故障或被攻击破坏，但整个系统的安全并不会受到威胁。例如，如果一个Web服务器的策略阻止修改它显示的文件，那么即使服务器被攻破了，TE策略也能阻止被攻破的服务器修改那些文件。这样就消除了通过Web服务器的漏洞攻击造成对网站的威胁。只有被攻破的应

用程序受到影响，并且它会被我们在策略中定义的访问权限制。

在具体的系统运行中，在SELinux系统启动的时候，会加载一个叫做policy.*的安全策略权限文件，这个文件中就定义了相关权限。如果用户在文件中设定了SELinux是不能在开机后转回permissive模式的话，那么系统的root用户则可能无法修改当中的设定，也就是说root在SELinux中已经不具有默认的所有权限。因此，即便是黑客盗取了root用户密码并成功入侵到用户的计算机，也只能在他入侵的这个自治域内进行破坏和信息窃取活动，并不会像以前那样扩散到整个Linux系统。因此，在具体的使用过程中，用户会深切地感觉到，在传统的Linux中，root拥有所有权限，可以执行任何操作；而在使用采用SELinux的Linux时发现自己即便是root，有些操作也不允许执行。

#### 3. 传统 Linux 与 SELinux 的区别

传统Linux与SELinux的最大区别就是：在传统的Linux自由访问控制（DAC）之后，SELinux在内核中使用强制访问控制机制（MAC）检查允许的操作。

在DAC下，文件客体的所有者提供了客体上的潜在风险控制。用户可以通过错误配置的chmod命令和一个非期望的访问权限传递，暴露一个文件或目录给一个恶意信任者。这个用户启动的进程，如CGI脚本，可在这个用户拥有的文件做任何的操作。DAC实际上仅有两个主要的用户分类：管理者和非管理者。为了解决权限分级，它使用了访问控制列表（Access Control Lists），给非管理者用户提供不同的权限。而root用户对文件系统有完全自由的控制权。

MAC访问控制框架可以定义所有的进程（称为主体）对系统的其他部分（如文件、设备、socket、端口和其他进程，它们被称为客体或目标对象）进行操作的权限或许可。这些许可由进程和客体的安全策略来定义，通过内核实现控制。这种方式可给一个进程仅授予操作所需要的权限，这遵循了最小权限原则。在MAC下，即使用户用chmod暴露了他们的数据，但进程还是无法修改在策略中没有许可的文件。

### 6.3.3 SELinux 中的上下文（context）

#### 1. SELinux 上下文核心元素

SELinux系统中的进程和文件都标记了SELinux的上下文（context），这个上下文包含了许多有用的信息，包括：SELinux用户、角色、类型、级别等。在运行SELinux的时候，这些上下文信息都被用来辅助进行访问控制决策。尤其是在目前最新的Fedora 10 Linux操作系统中，SELinux综合提供了RBAC、类型增强（Type Enforcement，简称TE）、多级别安全（Multi-Level Security，简称MLS）三种访问控制机制。

下面是一个SELinux上下文的例子。SELinux上下文广泛使用在进程、Linux用户、文件以及运行SELinux的Linux操作系统中。使用如下命令可以浏览文件（或者目录的）SELinux上下文：

```
#ls -Z yum.log
-rw-r--r--  root root system_u:object_r:root_t:s0     yum.log
```

SELinux上下文的组成为：

```
SELinux user:role:type:level
```

理解SELinux上下文，是正确和高效使用它的前提。因此，本节对它的组成进行一一介绍。

1）SELinux用户（SELinux user）

SELinux user标识一群被授权的角色或者是一个特定的MLS范围。每一个Linux系统用户都通过SELinux机制被映射为一个SELinux用户，这使得Linux用户可以继承SELinux用户的访问权限。这个标识主要用于限制用户可以进入的角色和级别范围。我们可以使用root用户的权限来运行semanage login –l命令，从而查看SELinux用户和Linux用户的映射关系：

```
# semanage login -l

登录名                 SELinux 用户           MLS/MCS 范围

__default__              unconfined_u           s0
root                    unconfined_u           s0-s0:c0.c1023
system_u                 system_u              s0-s0:c0.c1023
```

上述例子的运行结果是在Fedora 10 中获得的，采用上述命令的运行结果可能在不同版本的Linux中有所区别。从上面可以清楚地看到：Login Name列列出了Linux系统用户，而SELinux User列列出了这些系统用户所对应的SELinux用户。对于进程来说，这个SELinux用户直接限制了什么角色和级别可以为该用户访问。最后一列的MLS/MCS Range则给出了MLS和MCS（Multi-Category Security，多种类安全）机制所采用的级别（level）。关于级别，我们将在下面详细介绍。

2）角色（role）

SELinux中有一部分采用我们在本章中介绍的基于角色的访问控制（RBAC）机制。而role是RBAC机制中的一个属性。SELinux用户被授权为角色，而角色则被授权为对应的可访问域（domain）。因此，角色作为域和SELinux用户之间联系的媒介。通过角色可以决定SELinux用户可以进入哪些域，而最终决定了SELinux用户可以访问哪些对象类型。通过这种机制可以降低权限提升的风险。

3）类型（type）

类型是类型强制（Type Enforcement）机制的一个属性。这个类型定义了进程类型和文件类型。SELinux机制规则明确定义了类型间互相访问、域访问类型或者是域间相互访问的规则和许可。只有在某条SELinux机制规则允许的情况下，才允许上述的访问发生。

4）级别（level）

级别是上述MLS和MCS机制的另一个重要属性。一个MLS范围是一个级别对，采用区间标识，比如（最低级别，最高级别）或者（s0，s5）。每个级别都是一个种类敏感的数对，然而种类是可选的。如果存在种类，则可以写为sensitivity：种类。如果没有种类，则写为sensitivity即可。

如果种类集合是连续的，则在表示时可以进行简写。比如，c0.c2 与c0,c1,c2 表示同样的含义。举个例子，在Fedora 10 中，目标机制（Targeted Policy）对MCS进行了增强，因此它只有一个敏感级别s0。MCS支持 1024 个不同的种类，从c0 一直到 1023。因此，s0-s0:c0.c1023 所有的种类都被授权。

另外，与级别相关的/etc/selinux/targeted/setrans.conf文件非常重要，切忌使用诸如vi、gedit等文本编辑器对其进行直接编辑，可以使用semange命令进行修改，这样才能确保修改的正确性。

5）类型强制（Type Enforcement）

SELinux策略大部分都是一套声明和规则一起定义的类型强制（TE）策略，一个定义良好、严格的TE策略可能包括上千个TE规则，TE规则数量的巨大并不令人惊奇，因为它们表达了所有

由内核暴露出的允许对资源的访问权，这就意味着每个进程对每个资源的访问尝试都必须至少要有一条允许的TE访问规则，如果我们仔细思考一下现代Linux操作系统中进程和资源的数量，就明白为什么在策略中有那么多的TE规则了。当我们添加由TE规则控制的审核配置和标志时，对于具有严格限制的SELinux策略，常常会见到它包含有上千条规则，在第三篇中，我们将会讨论如何创建和管理这些大量的规则，现在，我们先理解一下TE规则是如何工作的。

TE规则的绝对数量对理解SELinux策略是一个挑战，但是规则本身并不复杂，它们的分类相对较少，所有的规则基本上都属于两类范畴：访问向量（AV）和类型规则。我们使用AV规则允许或审核两个类型之间的访问权，而在某些情况下使用类型规则控制默认的标记决定。

正如它们的名字暗示的意思，TE规则通过安全上下文与所有资源联合起来对类型起作用，策略语言包括了另外的允许我们定义类型及其策略组件的语句。

SELinux是不会管用户的，可以给同一个程序指定多个域类型（因此有不同的特权集），这样就允许引入角色的概念。因此，访问控制的标准仍然是基于程序的域类型而不是用户的特权。

#### 2．域转换

前面我们介绍过，SELinux一个很大的特点就是将进程和用户的执行权限限定在一个域（domain）内。因此，即使root用户也不可能具有太大的权限，从而保证了安全性。进程从一个域转换到另一个域需要通过执行一个具有新域的“入口点”权限的应用程序来实现。这个“入口点”许可在SELinux机制中使用，它用来控制某些应用程序可以用来进入一个域。为了清楚地说明这个问题，下面给出一个实际的例子加以说明。

一个用户想要改变他的用户密码。为了修改密码，我们知道，应该运行passwd程序。/usr/bin/passwd可执行命令的标记为passwd_exec_t类型，如下所示：

```
# ls -Z /usr/bin/passwd
-rwsr-xr-x  root root system_u:object_r:passwd_exec_t:s0 /usr/bin/passwd
```

在实际的执行过程中，该命令访问了/etc/shadow文件，该文件的类型为shadow_t，如下所示：

```
# ls -Z /etc/shadow
-r--------  root root system_u:object_r:shadow_t:s0    /etc/shadow
```

SELinux机制的相关规则规定：运行在passwd_t域的进程对标记为shadow_t类型的文件具有读和写的权限。并且，shadow_t类型仅仅只赋予和密码修改相关的那些文件，这些文件包括/etc/gshadow、/etc/shadow以及它们的备份文件。根据这个规则，用户就可以知道：passwd_t域的进程具有passwd_exec_t类型的“入口点”权限。因此，当一个用户运行/user/bin/passwd程序进行密码修改时，这个用户的shell进程就切换到了passwd_t域。由于运行在passwd_t域的进程对标记为shadow_t类型的文件具有读和写的权限，所以passwd进程可以访问/etc/shadow文件，并且更新其相应的用户密码。当然，在SELinux机制中，如果是默认情况下（没有相应规则的规定情况下），该进程是无法访问相应文件的。可以进一步对这个问题详细解释为：在传统的Linux中（未使用SELinux），passwd程序是可信任的，因而可以修改经过加密的影子密码文件（/etc/shadow）。所以，passwd程序执行它自己内部的安全策略，允许普通用户修改属于他们自己的密码，同时也允许root修改所有密码。为了执行这个密码修改操作，passwd程序需要有移动和重新创建shadow文件的能力。在标准Linux中，它具有这个特权，因为passwd程序可执行文件在执行时被加上了setuid位，它作为root用户（它能访问所有文件）被允许对密码进行修改操作。然而，许多程序都可以

作为root允许（实际上，所有程序都有可能作为root允许）。这就意味着任何程序（当以root身份运行时）都有可能修改shadow文件。因此，类型强制使用户能做的事情是确保只有passwd程序（或类似受信任的程序）可以访问shadow文件，而不管运行程序的用户是谁（包括root在内）。

在上述例子当中，除了SELinux的相关规则约定外，类型强制机制在很大程度上也确保了以下几个方面的前提条件。

- 只有标记为 passwd_exec_t 类型的应用程序（命令）执行才能进入 passwd_t 域，其他的命令执行都不允许进入。
- 只有一些授权的域，比如 passwd_t 才能对 shadow_t 类型的文件具有写的权限。即使其他的进程具有超级用户的权限，也不允许对 shadow_t 类型的文件具有写的权限，因为它们并不运行在 passwd_t 域中。
- 只有一些授权的域才能转换到 passwd_t 域。比如，sendmail 进程运行在 sendmail_t 域中，在该域中它们没有合理的理由和权限执行 passwd 命令，因此也就不能转换到 passwd_t 域。
- 运行在 passwd_t 域中的进程只对授权的类型具有读和写的权限，例如，标记为 shadow_t 和 etc_t 类型的文件。这就有效地阻止了 passwd 命令对其他文件的任意读和写权限，从而保证了安全。

### 3．SELinux 中进程的上下文

为了了解SELinux中进程的上下文信息，我们可以使用ps –eZ命令来查看进程的上下文信息，如下所示：

```
# ps -eZ | grep passwd
unconfined_u:unconfined_r:passwd_t:s0 2923 pts/2 00:00:00 passwd
```

上面的例子显示了/usr/bin/passwd命令（标记为passwd_exec_t类型）的上下文信息。值得注意的是：这个类型不但定义了该进程执行的域，也定义了它可以访问的文件的类型。

下面，再使用该命令来显示系统中运行的命令的SELinux上下文信息。下面是部分显示信息：

```
# ps -eZ
LABEL                              PID TTY          TIME CMD
system_u:system_r:init_t:s0          1 ?        00:00:02 init
system_u:system_r:kernel_t:s0        2 ?        00:00:00 kthreadd
system_u:system_r:kernel_t:s0        3 ?        00:00:00 migration/0
system_u:system_r:kernel_t:s0        4 ?        00:00:00 ksoftirqd/0
system_u:system_r:kernel_t:s0        5 ?        00:00:00 watchdog/0
system_u:system_r:kernel_t:s0        6 ?        00:00:00 events/0
system_u:system_r:kernel_t:s0        7 ?        00:00:00 khelper
system_u:system_r:kernel_t:s0       80 ?        00:00:00 kintegrityd/0
system_u:system_r:kernel_t:s0       82 ?        00:00:00 kblockd/0
system_u:system_r:kernel_t:s0       84 ?        00:00:00 kacpid
system_u:system_r:kernel_t:s0       85 ?        00:00:00 kacpi_notify
system_u:system_r:kernel_t:s0      146 ?        00:00:00 cqueue
system_u:system_r:kernel_t:s0      150 ?        00:00:01 ata/0
system_u:system_r:kernel_t:s0      151 ?        00:00:00 ata_aux
```

```
system_u:system_r:kernel_t:s0    153 ?        00:00:00 ksuspend_usbd
system_u:system_r:kernel_t:s0    158 ?        00:00:00 khubd
system_u:system_r:kernel_t:s0    161 ?        00:00:00 kseriod
system_u:system_r:kernel_t:s0    201 ?        00:00:00 pdflush
system_u:system_r:kernel_t:s0    202 ?        00:00:00 pdflush
system_u:system_r:kernel_t:s0    203 ?        00:00:00 kswapd0
```

4．SELinux 中用户的上下文

可以采用id –Z命令来显示与Linux用户相关的SELinux上下文信息，如下所示：

```
# id -Z
unconfined_u:unconfined_r:unconfined_t:s0
```

值得一提的是：在Fedora最新的版本Fedora 10 中，Linux用户的运行默认为未限制（unconfined）。这个SELinux上下文表明Linux用户被映射到unconfined_t域，它们的MLS范围为s0-s0（也就等同于s0）。

### 6.3.4　SELinux 中的目标策略（Targeted Policy）

1．目标策略原理

targeted策略是从strict示例策略衍生而来的，它的结构和组织几乎是一样的，但strict策略更趋向于最大化使用SELinux所有特性，为大部分程序提供强大的安全保护，而targeted策略的目标是隔离高风险程序。使用targeted策略的好处是，一方面可以向Linux系统添加大量的安全保护，同时又尽量少影响现有的用户程序。targeted策略主要集中于面向网络的服务（即那些暴露在外任意遭受黑客攻击的组件），targeted策略是RHEL和Fedora系统上标准的策略，因为它在增强安全性和减少对现有应用程序影响之间达到了一个很好的平衡。

如果安装了targeted示例策略，可以在/etc/selinux/targeted/src/policy/目录下看到它的源文件，从各个方面来看，targeted示例策略源与strict示例源都非常相似。

targeted示例策略和strict示例策略之间主要的差异是使用了无限制的域类型unconfined_t，并移除了所有其他用户域类型，如sysadm_t和user_t，这也意味着基本的角色结构也被移除了，所有用户都以角色system_r运行，几乎所有的用户运行的程序都以unconfined_t域类型执行。当然，无限制域和限制域都需要接受可执行和可写的内存检查。在默认情况下，运行在无限制域下的主体不能分配可写和可执行的内存，这个机制降低了系统遭受缓冲区溢出攻击（buffer overflow attack）的风险。当然，这些内存检查可以通过设置我们下面需要详细介绍的布尔变量来关掉，它使得SELinux策略可以在运行时得到修改。

用户可以在./domain/unconfined.te中找到unconfined域定义，注意在targeted示例策略中，strict策略文件admin.te和user.te不再位于./domains/目录下，这些文件为strict示例策略定义了各种各样的用户域，每一个都具有受限的特权，在targeted示例策略中，所有程序都以unconfined_t域类型运行，除非它们都明确地指定了域类型（因此称其为targeted），本质上unconfined域可以访问所有的SELinux类型，使它免除SELinux安全控制（因此成为unconfined）。在strict示例策略中，./domains/program/包括许多策略模块，每个模块代表一个或多个域类型和关联的类型，以及为特定程序制定的规则。在targeted示例策略中，这个目录包括的文件要少得多，这些就是目标。

目标示例策略模块与strict策略中策略模块类似，例如：strict ping模块和targeted ping模块是一致的，但有部分targeted模块只是简单地定义类型使域不受限制（不是targeted），例如：如果查看crond的targeted策略(crond.te)，会发现有一行unconfined_domain(crond_t)。这个宏在targeted示例中定义在./policy/macros/global_macros.te文件中，它将crond域类型提供了所有SELinux访问权，使得它不受限制，如果我们将其与strict版本（/etc/selinux/strict/src/policy/domains/program/crond.te）crond模块进行比较，会看到它们之间有很大的差异，在targeted策略中，crond被认为是不受限制的域，但在这两个策略中ping却保留为strict域。

2. 限制进程

几乎所有的服务进程都在限制下运行。并且，大多数以root身份运行的系统进程（比如说passwd进程）都是受限制的。当进程受限制时，它只能在自己限制的域内运行，例如Web服务进程httpd只能运行在httpd_t域内。如果一个受限制的进程被黑客攻击并控制了，根据SELinux策略配置，这个黑客也仅仅只能访问这个受限制的域，因此攻击所带来的危害也比传统的Linux小了很多。

以下通过一个具体的限制进程的例子（RHEL或者Fedora系统中的例子）来说明SELinux是如何将进程限制在自己的域内运行的。这个例子以用户非常熟悉且常用的Apache服务器中的httpd进程为例，来介绍SELinux是如何阻止httpd进程来访问由其他域管理的文件类型的。

（1）运行sestatus命令来确认Linux中SELinux是运行的，它运行在enforcing模式下（具体有关SELinux运行的三种模式，将在后面小节详细介绍，该模式可以简单理解为SELinux的完全运行模式，它可以进行强制访问控制），且确保采用了目标策略：

```
# sestatus
SELinux status:              enabled
SELinuxfs mount:             /selinux
Current mode:                enforcing
Mode from config file:       enforcing
Policy version:              23
Policy from config file:     targeted
```

上述运行结果表明SELinux运行在enforcing模式下，且采用了目标策略。

（2）采用Linux中的root用户权限，使用如下命令在httpd的工作目录中创建一个新的文件：

```
#touch /var/www/html/testfile
```

（3）运行如下命令来查看该文件的SELinux上下文信息：

```
# ls -Z /var/www/html/testfile
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/
html/testfile
```

从上述结果可以清楚地看到：在默认情况下，Linux用户是非限制的，因此刚创建的testfile文件的SELinux上下文中的类型标记为unconfined_u。RBAC访问控制机制是用于进程的，不是用于文件。并且，角色对于文件来说也没有什么太大的含义，因此上述结果中的object_r角色也仅仅是一个用于文件的通用角色。在/proc目录下，与进程相关的文件可以采用system_r角色。另外，结果中的httpd_sys_content_t类型允许httpd进程访问该文件。

（4）以Linux的root用户身份运行下述命令来运行httpd进程，如下所示：

```
# /sbin/service httpd start
启动 httpd:                                                    [确定]
```

（5）切换到一个Linux用户具有权限的目录下，运行如下命令，结果为该命令能够正确地执行并下载文件：

```
# wget http://localhost/testfile
--2011-05-06 06:46:26--  http://localhost/testfile
正在解析主机 localhost... 127.0.0.1
Connecting to localhost|127.0.0.1|:80… 已连接。
已发出 HTTP 请求，正在等待回应... 200 OK
长度：0 [text/plain]
Saving to: 'testfile'

    [ <=>                                  ] 0          --.-K/s   in 0s

2011-05-06 06:46:26 (0.00 B/s) - 'testfile' saved [0/0]
```

（6）使用chcon命令来对文件的类型进行重新标识。然而，这样的标识不是永久性的修改，一旦系统重启，该标识就会改变回去。对于文件类型的永久性改变，需要采用semanage命令，这个命令在后面将进行详细介绍。下面，以root用户的身份，运行如下命令来将上面步骤中创建的testfile文件的类型改为由Samba进程使用的文件：

```
# chcon -t samba_share_t /var/www/html/testfile
```

然后，运行ls -z /var/www/html/testfile命令来查看改变的结果，如下所示：

```
# ls -Z /var/www/html/testfile
-rw-r--r--  root root unconfined_u:object_r:samba_share_t:s0 /var/www/html/
testfile
```

（7）在传统的Linux中httpd进程可以访问testfile文件，下面需要尝试一下在SELinux中，该进程是否能够成功地访问testfile文件。如步骤（5）所示，再次运行该命令进行文件下载工作，发现命令运行失败，文件没有权限下载，运行结果如下所示：

```
# wget http://localhost/testfile
--2011-05-06 07:04:14--  http://localhost/testfile
正在解析主机 localhost… 127.0.0.1
Connecting to localhost|127.0.0.1|:80… 已连接。
已发出 HTTP 请求，正在等待回应… 403 Forbidden
2011-05-06 07:04:14 错误 403：Forbidden。
```

通过上述 7 个步骤的详细演示可以得知：虽然传统的Linux的DAC机制允许httpd进程访问testfile文件，然而SELinux的MAC机制却拒绝该访问操作。原因在于：该文件的类型（samba_share_t）httpd进程不能访问，因此SELinux拒绝了该操作。同时，SELinux对这些操作日志进行了详细的记载，以方便系统管理员事后进行审计和处理，可以查看/var/log/messages文件，如下所示：

```
May  6 07:04:14 wjl setroubleshoot: SELinux is preventing httpd (httpd_t) "getattr" to /var/www/html/testfile (samba_share_t). For complete SELinux messages. run sealert -l da48547d-9103-4a01-95ef-03fc1e48092e
May  6 07:04:14 wjl setroubleshoot: SELinux is preventing httpd (httpd_t) "getattr" to /var/www/html/testfile (samba_share_t). For complete SELinux messages. run sealert -l da48547d-9103-4a01-95ef-03fc1e48092e
```

另外，相关的错误日志也可以查看/var/log/audit/audit.log文件，如下所示：

```
type=AVC msg=audit(1241564654.246:26): avc:  denied  { getattr } for  pid=2744 comm="httpd" path="/var/www/html/testfile" dev=sda1 ino=471358 scontext
=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file
type=SYSCALL msg=audit(1241564654.246:26): arch=40000003 syscall=196 success
=no exit=-13 a0=b94aeea8 a1=bfb26bec a2=54dff4 a3=2008171 items=0 ppid=2741 pid=2744 auid=500 uid=48 gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=1 comm="httpd" exe="/usr/sbin/httpd" subj=unconfined_u:system
_r:httpd_t:s0 key=(null)
```

并且，由于该操作涉及httpd服务进程，由于该服务也有自己的日志文件，因此，也可以通过/var/log/httpd/error_log文件进行查看，如下所示：

```
[Wed May 06 07:04:14 2011] [error] [client 127.0.0.1] (13)Permission denied: access to /testfiledenied
```

### 3. 非限制进程

非限制的进程运行在非限制域中。比如，init进程运行在非限制的initrc_t域中，非限制的kernel进程运行在kernel_t域中，非限制的用户运行在unconfined_t域中。对于非限制的进程，SELinux策略规则仍然适用，然而有关允许进程运行在非限制域中的规则几乎允许所有的访问。因此，如果一个非限制进程被黑客控制的话，那么SELinux将不能阻止黑客获取对系统资源和数据的访问权限，当然DAC规则仍然适用，因为SELinux机制是在DAC层次上附加一层对Linux的增强，而不是简单地替代DAC。

下面将给出一个具体的例子来说明Apache Http服务器（httpd进程）在非限制的条件下运行，是如何访问本应由Samba服务器访问的数据的。值得注意的是：在Red Hat Linux和Fedora 10 中，httpd进程默认是限制运行在httpd_t域中的。下面的例子假设用户系统中安装了httpd、wget（一种类似于Windows系统下Flashget软件的多线程下载工具）、settroubleshoot-server、audit等工具包，并且SELinux机制运行在enforcing模式下。

（1）运行sestatus命令来确认Linux中SELinux是运行的，且运行在enforcing模式下，运行结果如下所示：

```
# sestatus
SELinux status:                enabled
SELinuxfs mount:               /selinux
Current mode:                  enforcing
```

```
Mode from config file:        enforcing
Policy version:              23
Policy from config file:      targeted
```

（2）以root用户身份，创建一个新的测试文件testfile2。该文件路径为httpd进程的工作目录下。

```
#touch /var/www/html/testfile2
```

（3）使用ls-z命令来查看新创建文件的SELinux上下文信息，如下所示：

```
# ls -Z /var/www/html/testfile2
-rw-r--r--      root   root   unconfined_u:object_r:httpd_sys_content_t:s0
/var/www/html/testfile2
```

从上述结果中可以清楚地看到：Linux用户默认运行在非限制域中，所以testfile2 文件上下文信息标识为unconfined_u。并且，RBAC用于进程，而不是文件。另外，对于文件来说角色也没有特别的含义，因此赋予其object_r为较通用的角色。而httpd_sys_content类型则允许httpd进程对该文件进行访问。

（4）采用chron命令暂时对文件的标识进行改变。同本章前面的例子一样，一旦系统重启，该改变将会失效。若要永久改变文件的标识，可以采用semanage命令进行操作。使用root用户身份，运行如下命令来将文件类型改为由Samba服务器可访问的类型：

```
#chcon -t samba_share_t /var/www/html/testfile2
```

然后，采用ls-z命令来查看修改后的结果：

```
# ls -Z /var/www/html/testfile2
-rw-r--r--  root root unconfined_u:object_r:samba_share_t:s0 /var/www/html/
testfile2
```

（5）在将httpd进程从限制修改为非限制之前，需要将该进程停止，使用如下命令：

```
#/sbin/service httpd stop
```

（6）以root用户身份运行如下命令，来改变httpd进程的类型，以将其从限制改为非限制：

```
#chcon -t unconfined_exec_t /usr/sbin/httpd
```

然后，使用ls-z命令来查看进程，以确保修改生效，如下所示：

```
# ls -Z /usr/sbin/httpd
-rwxr-xr-x  root root system_u:object_r:unconfined_exec_t:s0 /usr/sbin/httpd
```

（7）为了使httpd在运行时该修改生效，需要重新启动httpd进程，如下所示：

```
#/sbin/service httpd start
```

（8）采用ps-ez | grep httpd命令来查看httpd进程运行在非限制域中的情况，如下所示：

```
# ps -eZ | grep httpd
unconfined_u:system_r:unconfined_t:s0 2921 ?  00:00:00 httpd
unconfined_u:system_r:unconfined_t:s0 2923 ?  00:00:00 httpd
unconfined_u:system_r:unconfined_t:s0 2924 ?  00:00:00 httpd
unconfined_u:system_r:unconfined_t:s0 2925 ?  00:00:00 httpd
unconfined_u:system_r:unconfined_t:s0 2926 ?  00:00:00 httpd
```

```
unconfined_u:system_r:unconfined_t:s0 2927 ?  00:00:00 httpd
unconfined_u:system_r:unconfined_t:s0 2928 ?  00:00:00 httpd
unconfined_u:system_r:unconfined_t:s0 2929 ?  00:00:00 httpd
unconfined_u:system_r:unconfined_t:s0 2930 ?  00:00:00 httpd
unconfined_u:system_r:unconfined_t:s0 2931 ?  00:00:00 httpd
```

（9）在有权限许可的目录下，运行如下命令来测试httpd进程运行在非限制情况下的效果，如下所示：

```
# wget http://localhost/testfile2
--2011-05-06 07:24:47--  http://localhost/testfile2
正在解析主机 localhost… 127.0.0.1
Connecting to localhost|127.0.0.1|:80… 已连接。
已发出 HTTP 请求，正在等待回应… 200 OK
长度：0 [text/plain]
Saving to: 'testfile2'


   [ <=>                                    ] 0          --.-K/s   in 0s


2011-05-06 07:24:47 (0.00 B/s) - 'testfile2' saved [0/0]
```

上述运行结果显示，该命令能够成功运行。虽然httpd进程原来没有访问标记为samba_share_t类型文件的权限，然而由于修改，httpd进程现在运行在非限制环境下（unconfined_t），所以SELinux转为执行DAC机制，而wget命令可以访问该文件，所以成功执行。

（10）测试结束后，需要使用如下命令来恢复该httpd进程本来的运行限制：

```
#restorecon -v /user/sbin/httpd
```

然而，采用如下命令来进行查看，发现httpd进程又从非限制状态恢复为限制状态：

```
# ls -Z /usr/sbin/httpd
-rwxr-xr-x  root root system_u:object_r:httpd_exec_t:s0 /usr/sbin/httpd
```

（11）使用如下命令重新启动httpd，以使修改在httpd进程运行时生效，并通过ps-ez命令来查看进程的运行状态：

```
#/sbin/service httpd restart
# ps -eZ | grep httpd
unconfined_u:system_r:httpd_t:s0 2963 ?      00:00:00 httpd
unconfined_u:system_r:httpd_t:s0 2965 ?      00:00:00 httpd
unconfined_u:system_r:httpd_t:s0 2966 ?      00:00:00 httpd
unconfined_u:system_r:httpd_t:s0 2967 ?      00:00:00 httpd
unconfined_u:system_r:httpd_t:s0 2968 ?      00:00:00 httpd
unconfined_u:system_r:httpd_t:s0 2969 ?      00:00:00 httpd
unconfined_u:system_r:httpd_t:s0 2970 ?      00:00:00 httpd
unconfined_u:system_r:httpd_t:s0 2971 ?      00:00:00 httpd
unconfined_u:system_r:httpd_t:s0 2972 ?      00:00:00 httpd
```

```
unconfined_u:system_r:httpd_t:s0 2973 ?       00:00:00 httpd
```

到此为止，上述例子详细地演示了SELinux是如何通过限制进程的运行来保证Linux系统安全的，也看到了如果将进程改为unconfined运行状态的安全隐患，因此提醒广大用户在实际应用过程中切忌不要随便将进程默认的限制运行状态改为非限制状态，以免给Linux系统带来不必要的麻烦和安全隐患。

4．限制和非限制用户

在前面的介绍中曾经提到过，每个传统的Linux用户在SELinux中都被映射为一个SELinux用户。这使得Linux用户能够继承在SELinux用户上的访问控制。用户可以使用semanage login-l命令来查看两类用户之间的具体映射情况，如下所示：

```
# semanage login -l

登录名                   SELinux 用户           MLS/MCS 范围

__default__               unconfined_u            s0
root                     unconfined_u           s0-s0:c0.c1023
system_u                  system_u              s0-s0:c0.c1023
```

举个例子，在Fedora 10 中，Linux用户默认被映射到SELinux的_default_login中，从而映射为unconfined_u用户类型。

下面通过一个更加具体的在SELinux中添加新用户的例子，来说明SELinux是如何来映射Linux用户的。该例子的具体操作步骤如下。

（1）以root用户身份，运行useradd命令，添加一个新用户liyang：

```
#useradd liyang
```

（2）以root用户身份，运行passwd命令，修改该用户的密码：

```
#passwd liyang
```

（3）退出当前的root运行身份，以liyang的身份重新登录Linux。重新登录后，SELinux将为用户liyang生成SELinux上下文，查看该用户的上下文：

```
# id -Z
unconfined_u:unconfined_r:unconfined_t:s0
```

可以清楚地看到，当Linux添加一个新用户时，SELinux将默认地映射该用户为unconfined_u类型，角色为unconfined_r，以及unconfined_t级别。

限制和非限制用户都需要接受可执行和可写的内存检查，并且受MCS和MLS机制的约束。如果一个非限制的用户执行了一个从unconfined_t域向一个允许的域转变的应用程序，非限制用户仍要接受那个转变到的域的限制。这个就保证了即使一个用户是非限制的，这个应用也是受限的，因此，软件的漏洞所引起的风险就能得到限制。

### 6.3.5　SELinux 配置文件和策略目录介绍

1．SELinux 主配置文件——/etc/selinux/config

SELinux主配置文件/etc/selinux/config控制系统下一次启动过程中载入哪个策略，以及系统

运行在哪个模式下，可以使用sestatus命令确定当前SELinux的状态，下面显示了一个config文件的例子：

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
# enforcing - SELinux security policy is enforced.
# permissive - SELinux prints warnings instead of enforcing.
# disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
#                 targeted - Targeted processes are protected.
#                 mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

这个文件控制两个配置设置——SELinux模式和活动策略。

- SELinux 模式（由上述例子中第 6 行的 SELINUX 选项确定）：可以被设置为 enforcing、permissive 或 disabled 三种。
  - 在 enforcing 模式下，策略被完整执行，这是 SELinux 的主要模式，应该在所有要求增强 Linux 安全性的操作系统上使用。
  - 在 permissive 模式下，策略规则不被强制执行，相反，只是审核遭受拒绝的消息，除此之外，SELinux 不会影响系统的安全性，这个模式在调试和测试一个策略时非常有用。
  - 在 disabled 模式下，SELinux 内核机制是完全关闭了的，只有系统启动时策略载入前系统才会处于 disabled 模式，这个模式和 permissive 模式有所不同，permissive 模式有 SELinux 内核特征操作，但不会拒绝任何访问，只是进行审核，在 disabled 模式下，SELinux 将不会有任何动作，只有在极端环境下才使用这个模式，例如，当策略错误阻止用户登录系统时，即使在 permissive 模式下也有可能发生这种事情，或我们不想使用 SELinux 时。这里需要特别注意的是：在 enforcing 和 permissive 模式或 disabled 模式之间切换时要小心，当返回 enforcing 模式时，通常会导致文件标记不一致。SELinux 配置文件中的模式设置由 init 使用，在它载入初始策略前配置 SELinux 使用。
- SELinux 活动策略：SELinux 配置文件中的 SELINUXTYPE 选项告诉 init 在系统启动过程中载入哪个策略，这里设置的字符串必须匹配用来存储二进制策略版本的目录名，例如，我们使用 MLS 策略作为例子，因此我们设置 SELINUXTYPE=MLS，确保我们想要内核使用的策略在/etc/selinux/config 文件中。在上述例子中，活动策略默认为目标策略（targeted），由第 10 行给出。

### 2．SELinux 策略目录

SELinux系统上安装的每个策略在/etc/selinux/目录下都有它们自己的目录，子目录的名字对应于策略的名字（如strict、targeted、refpolicy等），在SELinux配置文件中就要使用这些子目录名字，告诉内核在启动时载入哪个策略，在本章中提到的所有路径都是相对域策略目录路径/etc/selinux/[policy]/的，下面是Fedora 10 系统上/etc/selinux/目录的简单列表输出：

```
# ls -lZ /etc/selinux/
-rw-r--r--  root root system_u:object_r:selinux_config_t:s0 config
-rw-------  root root system_u:object_r:selinux_config_t:s0 restorecond.conf
-rw-r--r--  root root system_u:object_r:selinux_config_t:s0 semanage.conf
drwxr-xr-x  root root system_u:object_r:selinux_config_t:s0 targeted
```

正如上面所看到的，在我们的系统上安装了一个策略目录：targeted。注意目录和策略子目录都是用selinux_config_t类型进行标记的。

semodule和semanage命令管理策略的许多方面，semodule命令管理可载入策略模块的安装、更新和移除，它对可载入策略包起作用，包括一个可载入策略模块和文件上下文消息，semanage工具管理添加、修改和移除用户、角色、文件上下文、多层安全（MLS）/多范畴安全（MCS）转换、端口标记和接口标记，关于这些工具的更多信息在它们的帮助手册中。

每个策略子目录包括的文件和文件如何标记必须遵守一个规范，这个规范被许多系统实用程序使用，帮助管理策略，通常，任何设计优良的策略源树都将正确安装策略文件，下面是targeted策略目录的列表输出，它就是一个典型：

```
# ls -lZ /etc/selinux/targeted/
drwxr-xr-x  root root system_u:object_r:default_context_t:s0 contexts
drwxr-xr-x  root root system_u:object_r:selinux_config_t:s0 modules
drwxr-xr-x  root root system_u:object_r:policy_config_t:s0 policy
-rw-r--r--  root root system_u:object_r:selinux_config_t:s0 setrans.conf
-rw-r--r--  root root system_u:object_r:selinux_config_t:s0 seusers
```

如上所示，一个正在运行的系统不需要src/目录，它包括了安装的策略源树，要么是示例策略，要么是应用策略源树，实际上二进制策略文件存储在.policy/目录中的policy.[ver]文件中，这里的[ver]就是策略二进制文件的版本号，如policy.19，这就是系统启动时载入内核的文件。

### 6.3.6　使用 SELinux 的准备

#### 1．SELinux 相关的安装包

虽然在有的Linux发行套件中已经默认安装了SELinux（例如Fedora 10），然而用户还是需要了解具体安装SELinux所需要的安装包，下面对其进行简要介绍。主要包括以下几个部分。

- Policycoreutils：提供与 SELinux 相关的命令，比如 semanage、restorecon、audit2allow、semodule、load_policy，以及 setsebool 等，来操作和管理 SELinux。
- Policycoreutils-gui：提供图形化的工具 system-config-selinux 来管理 SELinux。
- Selinux-policy：提供 SeLinux 应用策略。该应用策略包括了所有的 SELinux 策略，并作为其他诸如目标策略（targeted policy）的基础使用。
- Selinux-policy-policy：提供 SELinux 策略。对于目标策略，安装 selinux-policy-targeted 包，对于 MLS 策略，则安装 selinux-policy-mls 包。需要说明的是：在 Fedora 8 中，strict 策略与目标策略结合在一起。
- Setroubleshoot-server：翻译 SELinux 拒绝操作信息，成为 sealert 软件可以查看的详细描述信息。

- Setools、setools-gui 和 setools-console：这些安装包提供了与 SELinux 有关的策略分析和检索、审计日志监控、文件上下文管理等命令和工具。
- Libselinux-utils：提供诸如 avsstat、getenforce、getsebool、matchpathcon、selinuxconlist、selinuxdefcon、selinuxenabled、setenforce、togglesebools 等工具。
- Mcstrans：提供对 SELinux 上下文中级别（比如 s0-s0:c0.c1023）信息的翻译工作，在默认情况下该软件包不安装。

### 2．与 SELinux 有关的日志文件

SELinux有许多相关的日志文件来记录在运行过程中对操作的拒绝日志，以便用户在后续过程中进行审计评估。在默认情况下，SELinux将拒绝日志写入到/var/log/audit/audit.log文件中。该文件的部分内容显示如下：

```
type=USYS_CONFIG msg=audit(1236119424.663:9): user pid=2074 uid=0 auid
=4294967295 ses=4294967295subj=system_u:system_r:firstboot_t:s0 msg
='changing system time: exe="/sbin/hwclock" (hostname=?, addr=?, terminal
=console res=success)'
type=USER_AUTH msg=audit(1236119439.103:10): user pid=2078 uid=0 auid
=4294967295 ses=4294967295 subj=system_u:system_r:firstboot_t:s0 msg
='op=PAM:authentication acct="root" exe="/usr/sbin/userhelper" (hostname=?,
addr=?, terminal=console res=success)'
```

另外，如果setroubleshooted运行的话，在/var/log/audit/audit.log中的记录将被翻译成容易理解和阅读的方式，保存在/var/log/messages文件中：

```
Mar  4 06:24:26 wjl acpid: starting up
Mar  4 06:24:36 wjl pcscd: pcscdaemon.c:498:main() pcsc-lite 1.4.102 daemon ready.
Mar  4 06:24:36 wjl acpid: client connected from 1723[68:68]
Mar  4 06:24:37 wjl NetworkManager: <info>  starting...
Mar  4 06:24:37 wjl NetworkManager: <WARN>  nm_generic_enable_loopback():
error -17 returned from rtnl_addr_add():#012Sucess#012
Mar  4 06:24:37 wjl NetworkManager: <info>  eth0: driver is 'pcnet32'.
Mar  4 06:24:37 wjl NetworkManager: <info>  Found new Ethernet device 'eth0'.
Mar  4 06:24:37 wjl NetworkManager: <info>  (eth0): exported as /org/
freedesktop/Hal/devices/net_00_0c_29_05_16_eb
Mar  4 06:24:37 wjl NetworkManager: <info>  Trying to start the supplicant…
Mar  4 06:24:37 wjl NetworkManager: <info>  Trying to start the system settings
daemon…
Mar  4 06:24:38 wjl nm-system-settings: Loaded plugin ifcfg-fedora: (c) 2007
- 2008 Red Hat, Inc.  To report bugs please use the NetworkManager mailing list.
Mar  4 06:24:41 wjl kernel: eth0: link up
```

当然，拒绝信息被送到不同的地方，这要根据不同的守护进程而定，表 6-2 列出了对应于不同的守护进程的日志文件的路径。

表 6-2　SELinux 日志文件列表

| 守护进程 | 日志文件 |
| --- | --- |
| auditd on | /var/log/audit/audit.log |
| Auditd off; rsyslogd on | /var/log/messages |
| Setroubleshootd, rsylogd, auditd on | /var/log/audit/audit.log 翻译后存入/var/log/messages |

为了启动上述的守护进程，需要分别配置auditd、rsyslogd以及setroubleshootd来使它们在系统启动时自动运行，可以root身份运行下述命令：

```
#/sbin/chkconfig -levels 2345 auditd on
#/sbin/chkconfig -levels 2345 rsyslogd on
#/sbin/chkconfig -levels 2345 setroubleshootd on
```

并且可以使用以下命令检查这些守护进程是否正常运行：

```
#/sbin/service auditd status
#/sbin/service rsyslogd status
#/sbin/service setroubleshootd status
```

### 3．启动和禁用 SELinux

启动和禁用SELinux的步骤非常简单，只需要修改其配置文件，然后执行重启即可。

以下是启动SELinux的详细步骤。

（1）编辑配置文件。

根据前面的介绍，编辑确定SELinux的运行模式和活动策略（见下面配置文件中的黑体部分）即可：

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
# enforcing - SELinux security policy is enforced.
# permissive - SELinux prints warnings instead of enforcing.
# disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
# targeted - Targeted processes are protected.
# mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

（2）使用getenforce命令和sestatus命令查看SELinux的运行模式和详细上下文信息：

```
#/usr/sbin/getenforce


#/usr/sbin/sestatus
```

（3）使用root用户身份重启系统即可：

```
#reboot
```

同理，禁用SELinux的详细步骤如下。

（1）编辑配置文件。

根据前面的介绍，编辑确定SELinux的运行模式和活动策略（见下面配置文件中的黑体部分）即可：

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
# enforcing - SELinux security policy is enforced.
# permissive - SELinux prints warnings instead of enforcing.
# disabled - No SELinux policy is loaded.
SELINUX=diabled
# SELINUXTYPE= can take one of these two values:
# targeted - Targeted processes are protected.
# mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

（2）使用getenforce命令和sestatus命令查看SELinux的运行模式和详细上下文信息：

```
#/usr/sbin/getenforce

#/usr/sbin/sestatus
```

（3）使用root用户身份重启系统即可：

```
#reboot
```

### 6.3.7 SELinux 中布尔（boolean）变量的使用

在SELinux中，有一个非常重要的概念称为布尔变量，它通常用来在运行时改变SELinux的部分策略，而不需要重新定义和改写策略文件。并且，这些布尔变量在很大程度上也与许多网络服务的访问权限相关，比如对网络文件系统（NFS）的访问、对FTP的访问，等等。

#### 1. 列出和查看系统布尔变量

通过列出系统中的布尔变量，可以知道与布尔变量相关的策略的开关情况（on或者是off状态），从而知道相关网络服务的访问权限状态。可以使用以下命令进行罗列：

```
# /usr/sbin/semanage boolean -l
SELinux 布尔值                    描述


ftp_home_dir -> 关    Allow ftp to read and write files in the user home
directories
xen_use_nfs                   -> 关   Allow xen to manage nfs files
xguest_connect_network        -> 关   Allow xguest to configure Network Manager
spamassassin_can_network    -> 关   Allow user spamassassin clients to use the
network.
httpd_can_network_relay       -> 关   Allow httpd to act as a relay
openvpn_enable_homedirs       -> 关   Allow openvpn to read home directories
```

```
 allow_execheap              -> 关   Allow unconfined executables to make their
heap memory executable.  Doing this is a really bad idea. Probably indicates a
badly coded executable, but could indicate an attack. This executable should
be reported in bugzilla
 httpd_can_network_connect_db   -> 关    Allow HTTPD scripts and modules to
connect to databases over the network.
 allow_ftpd_full_access        -> 关   Allow ftp servers to login to local users
and read/write all files on the system, governed by DAC.
 samba_domain_controller     -> 关   Allow samba to act as the domain controller,
add users, groups and change passwords.
 httpd_enable_cgi             -> 关   Allow httpd cgi support
 virt_use_nfs                 -> 关   Allow virt to manage nfs files
 allow_daemons_use_tty       -> 开   Allow all daemons the ability to read/write
terminals
```

上述结果中的SELinux boolean列列出了系统中的布尔变量名称，Description列则给出了该布尔变量的开关状态。比如，以下的布尔变量表示的意思是阻止FTP用户访问用户目录：

```
 ftp_home_dir  -> 关   Allow ftp to read and write files in the user home
directories
```

以下命令可以仅仅给出布尔变量的开关状态，而不给出它们的具体描述信息：

```
#/usr/sbin/getsebool -a
allow_console_login --> off
allow_cvs_read_shadow --> off
allow_daemons_dump_core --> on
allow_daemons_use_tty --> on
allow_domain_fd_use --> on
allow_execheap --> off
allow_execmem --> off
allow_execmod --> off
allow_execstack --> on
allow_ftpd_anon_write --> off
allow_ftpd_full_access --> off
allow_ftpd_use_cifs --> off
allow_ftpd_use_nfs --> off
allow_gssd_read_tmp --> on
allow_guest_exec_content --> off
allow_httpd_anon_write --> off
allow_httpd_mod_auth_ntlm_winbind --> off
allow_httpd_mod_auth_pam --> off
allow_httpd_sys_script_anon_write --> off
```

```
allow_kerberos --> off
allow_mount_anyfile --> on
allow_mplayer_execstack --> off
allow_nfsd_anon_write --> off
allow_nsplugin_execmem --> on
allow_polyinstantiation --> off
allow_postfix_local_write_mail_spool --> on
allow_ptrace --> off
allow_rsync_anon_write --> off
allow_saslauthd_read_shadow --> off
allow_smbd_anon_write --> off
allow_ssh_keysign --> off
allow_staff_exec_content --> on
allow_sysadm_exec_content --> on
allow_unconfined_exec_content --> on
allow_unconfined_mmap_low --> off
allow_unconfined_nsplugin_transition --> on
allow_unconfined_qemu_transition --> off
allow_user_exec_content --> on
allow_user_postgresql_connect --> off
allow_write_xshm --> off
allow_xguest_exec_content --> off
allow_xserver_execmem --> on
allow_ypbind --> off
allow_zebra_write_config --> on
browser_confine_xguest --> off
browser_write_xguest_data --> off
cdrecord_read_content --> off
exim_can_connect_db --> off
exim_manage_user_files --> off
exim_read_user_files --> off
fcron_crond --> off
ftp_home_dir --> off
global_ssp --> off
gpg_agent_env_file --> off
httpd_builtin_scripting --> on
httpd_can_network_connect --> off
httpd_can_network_connect_db --> off
httpd_can_network_relay --> off
```

并且，通过该命令可以查看具体的布尔变量（一个或者多个）的开关状态信息，如下所示：

```
#/usr/sbin/getsebool allow_console_login

#/usr/sbin/getsebool allow_console_login allow_cvs_read_shadow
```

### 2．配置布尔变量

使用setsebool命令可以设置布尔变量（改变其开关状态）。下面给出具体的例子来对其进行详细介绍。该例子表明如何来对httpd_can_network_connect_db布尔变量来进行状态设置，该布尔变量控制httpd服务器进程能否连接后台数据库系统。

（1）使用getsebool命令查看该布尔变量的默认状态，表明该变量的默认状态是off，即不允许httpd进程访问后端数据库服务器：

```
# /usr/sbin/getsebool httpd_can_network_connect_db
httpd_can_network_connect_db --> off
```

（2）使用setsebool命令改变该布尔变量的状态，从而使得httpd进程能够访问数据库服务器：

```
#/usr/sbin/setsebool httpd_can_network_connect_db on
```

改变成功后，可以继续使用getsebool命令进行查看：

```
#/usr/sbin/getsebool httpd_can_network_connect_db
```

（3）上述状态改变只是暂时性的，一旦系统重启，该变量状态将改变回初始状态，因此，可以使用以下命令永久性改变状态：

```
#/usr/sbin/setsebool -P httpd_can_network_connect_db on
```

### 3．一些与NFS和CIFS文件系统相关的布尔变量

对于大多数网络服务来说，各种网络服务默认情况下都能使用具有各自的服务类型的文件，比如NFS服务可以使用nfs_t类型的文件，samba服务可以使用cifs_t类型的文件。然而，由于NFS和CIFS这两个共享文件系统的特殊性，HTTP服务和FTP服务都不能使用标记这两个文件系统类型的文件，因此，在实际使用过程中，用户需要对控制它们的布尔变量进行显式的状态修改，如下所示。

（1）对于Apache HTTP服务器。

允许访问NFS文件系统，需要运行以下命令，修改布尔变量：

```
#/usr/sbin/setsebool -P httpd_use_nfs on
```

允许访问Samba文件系统，需要运行以下命令，修改布尔变量：

```
#/usr/sbin/setsebool -P httpd_use_cifs on
```

（2）对于Samba服务。

允许共享NFS文件系统的文件，需要运行以下命令，修改布尔变量：

```
#/usr/sbin/setsebool -P sambe_share_nfs on
```

（3）对于FTP服务器。

允许访问NFS文件系统，需要运行以下命令，修改布尔变量：

```
#/usr/sbin/setsebool -P allow_ftpd_use_nfs on
```

允许访问Samba文件系统，需要运行以下命令，修改布尔变量：

```
#/usr/sbin/setsebool -P llow_ftpd_use_cifs on
```

（4）对于其他服务。

可以使用以下命令来查找相关的布尔变量并根据上述的例子进行同样的修改即可。

查找与NFS相关的布尔变量：

```
#/usr/sbin/semanage boolean -l | grep nfs
```

查找与Samba相关的布尔变量：

```
#/usr/sbin/semanage boolean -l | grep cifs
```

# 第 7 章 紧密布控：企业 Web 服务器安全防护

## 本章导读

企业 Web 服务器是企业非常重要的“脸面”。很多企业都通过部署 Web 服务器，使用 Web Portal 来向客户展现企业风采和开展线上业务，作为世界 500 强企业来说更是如此。如何在保证安全的前提下来进行企业风采展现和线上业务开展，是企业需要认真考虑和权衡的事情。毕竟，一旦出现问题，将会给企业带来很大的负面影响，影响客户信任度和对企业的认可度。因此，本章将详细介绍 500 强企业如何来紧密布控，全面地保证开源企业 Web 服务器运维安全。

## 7.1 Web 安全威胁分析及解决思路

企业Web服务器主要面临以下几种安全威胁。

- 使用 HTTP 协议进行的拒绝服务攻击：攻击者会通过某些手段使服务器拒绝对 HTTP 应答。这样会使 Web 服务对系统资源（CPU 时间和内存）需求巨增，造成系统变慢甚至完全瘫痪，从而引起 HTTP 服务的中断或者合法用户的合法请求得不到及时的响应。
- 被攻击者获得 root 权限，威胁系统安全：由于开源 Web 服务器一般以 root 权限运行，攻击者通过它获得 root 权限，进而控制整个 Web 服务器系统。
- 由于服务器配置文件设置不当引起的安全问题：恶意者可以随意下载、修改或删除系统文件。这主要涉及到对访问者的内容和权限的限制。
- 通信安全：由于 Web 服务使用默认的 HTTP 协议进行通信，而通信的内容均采用明文方式，因此，客户与企业 Web 服务器的通信流量在 Internet 上进行传输，其重要内容皆可能遭受窃听和篡改。

针对上面几种安全威胁，本章将根据表 7-1 所示的思路来进行分析介绍。

表 7-1　Web 防护工作思路

| 主要防护手段 | 解决的主要安全威胁 | 安全威胁牵涉到的人员及操作 |
|---|---|---|
| Web 安全配置 | 避免拒绝服务攻击，因配置不当引起的安全问题 | 外部黑客的恶意攻击 |
| Web 访问控制和认证 | 避免企业内部用户/外部用户对资源的越权访问 | 外部黑客以及内部用户的越权访问 |
| SSL 安全通信 | 保证企业 Web 服务器与客户的通信安全，防止敏感、机密信息泄露 | 外部黑客、内部用户窃听、篡改相关重要通信数据 |
| 日志管理和流量分析 | 通过对企业内部用户及外部黑客在系统中运行产生相应的日志记录进行审计，发现安全问题和蛛丝马迹 | 用户、黑客在系统中的运行产生日志记录 |
| 其他网络架构和 chroot 安全措施 | 从架构上保证 Web 服务的运行安全，并减少 root 用户权限对 Web 服务操作的影响 | 外部黑客攻击、root 用户在 Web 服务操作方面的滥用 |

## 7.2 Web 服务器选型

### 7.2.1 HTTP 基本原理

Web服务器也称为WWW服务器或HTTP服务器（HTTP Server），它是Internet上最常见也是使用最频繁的服务器之一。Web服务器能够为用户提供网页浏览、论坛访问等服务。比如，用户在使用浏览器访问http://www.sina.com的时候，实际上就是在访问新浪的Web服务器，从该Web服务器获取需要的论坛资料和网页。

一个Web服务器之所以也被称为HTTP服务器，是因为它通过HTTP协议与客户端通信。这个

客户端通常指的是Web浏览器。HTTP 是一种让Web服务器与浏览器（客户端）通过Internet发送与接收数据的协议。它是一个请求、响应协议：客户端发出一个请求，服务器响应这个请求。HTTP运用可靠的TCP连接，通常用的TCP为 80 端口。它的第一个版本是HTTP/0.9，然后被HTTP/1.0取代。当前的版本是HTTP/1.1，由RFC2616 定义（RFC是Request For Comment的缩写，是由IETF管理，所有关于Internet的正式标准都以文档出版，但不是所有的RFC都是正式的标准，很多RFC的目的只是为了提供信息。RFC每一篇都用一个数字来标识（如RFC2401）数字越大说明RFC的内容越新。RFC是免费公开的，任何人都可以写RFC并提交IETF，一旦正式通过就可以正式发布，一旦发布RFC内容将不能再作任何修改，以后的修改只能通过新的RFC来处理，因此可以看到有很多新的RFC文档obsolete（废除）或update（更新）老的RFC）。

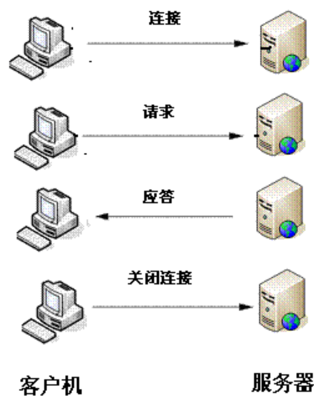


图 7-1　HTTP 基本原理示意

在HTTP中，客户端总是通过建立一个连接与发送一个HTTP请求来发起一个事务。服务器不能主动去与客户端联系，也不能给客户端发出一个回叫连接。客户端与服务器端都可以提前中断一个连接。例如，当用一个浏览器下载一个文件时，用户可以通过单击“停止”按钮来中断文件的下载，关闭与服务器的HTTP连接。

HTTP协议的工作原理主要包括以下四个步骤（参见图 7-1）。

（1）连接：Web浏览器与Web服务器建立连接，打开一个称为socket（套接字）的虚拟文件，此文件的建立标志着连接建立成功。

（2）请求：Web浏览器通过socket向Web服务器提交请求。HTTP的请求一般是GET或POST命令（POST用于FORM参数的传递）。GET命令的格式为：GET 路径/文件名 HTTP/1。其中，文件名指出所访问的文件，HTTP/1.0 指出Web浏览器使用的HTTP版本。

（3）应答：Web浏览器提交请求后，通过HTTP协议传送给Web服务器。Web服务器接收到后，进行事务处理，处理结果又通过HTTP传回Web浏览器，从而在Web浏览器上显示出所请求的页面。例如，假设客户机与www.mycompany.com:8080/mydir/index.html建立了连接，就会发送GET命令：GET /mydir/index.html HTTP/1.0。主机名为www.mycompany.com的Web服务器从它的文档空间中搜索子目录mydir的文件index.html。如果找到该文件，Web服务器把该文件内容传送给相应的Web浏览器。为了告知 Web浏览器传送内容的类型，Web服务器首先传送一些HTTP头信息，然后传送具体内容（即HTTP体信息），HTTP头信息和HTTP体信息之间用一个空行分开。其中，常用的HTTP头信息如下。

- HTTP 1.0 200 Ok：这是 Web 服务器应答的第一行，列出服务器正在运行的 HTTP 版本号和应答代码。代码“200 Ok”表示请求完成。
- MIME_Version:1.0：它指示 MIME 类型的版本。
- content_type:类型：这个头信息非常重要，它指示 HTTP 体信息的 MIME 类型。如 content_type:text/html 指示传送的数据是 HTML 文档。
- content_length:长度值：它指示 HTTP 体信息的长度（字节）。

（4）关闭连接：当应答结束后，Web浏览器与Web服务器必须断开，以保证其他Web浏览器能够与Web服务器建立连接。

### 7.2.2　为何选择 Apache 服务器

由于用户在通过Web浏览器访问信息资源的过程中，无须再关心一些技术性的细节，而且界面非常友好，因而Web在Internet上一推出就得到了爆炸性的发展。现在Web服务器已经成为Internet上最大的计算机群，Web文档之多、连接的网络之广，也令人难以想象。因此，Web服务器软件的数量也开始增加，Web服务器软件市场的竞争也越来越激烈。本章要介绍的就是一款最常用的Web服务器软件——Apache。

Netcraft是一家英国的互联网服务公司，提供互联网安全服务，以及对互联网的数据进行搜集和调查研究。Netcraft近日公布了 2010 年 2 月份的全球Web服务器使用情况调查报告。此次调查总共统计了 207 316 960 个站点的信息。2 月期间Apache主机名称数量经历了 1600 万的增长，排在增长趋势的第一位。增长第二位的是微软的Web服务器，主机名称增长数量为 1100 万。如图 7-2 和图 7-3 所示。

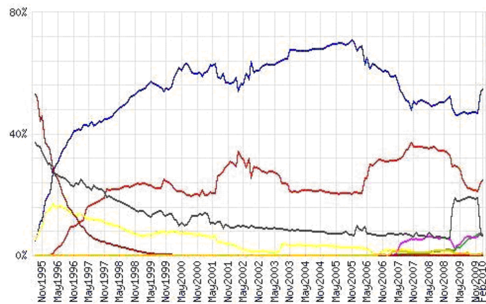



图 7-2　Netcraft 公司提供的 Web 服务器使用情况统计

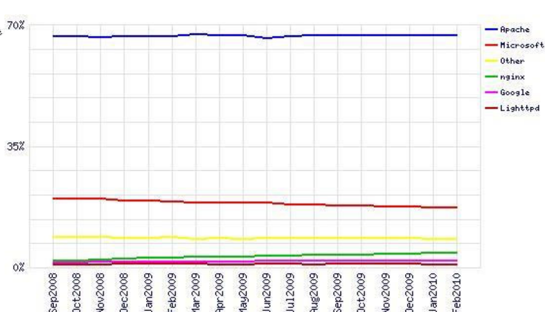



图 7-3　Netcraft 公司提供的繁忙度前 100 万位站点的 Web 服务器分布百分比

在Web服务器发展的初期，美国构架超级计算应用中心（NCSA）在 1995 年创建了当时一流的Web服务器。然而，NCSA Web服务器的主要开发人员后来几乎同时离开了NCSA，使得这个服务器项目停顿下来。与此同时，那些使用NCSA Web服务器的人们开始交换他们用于该服务器的补丁程序，他们也很快认识到处理管理这些补丁程序的论坛的重要意义。就这样，诞生了Apache Group。这一团体使用NCSA Web服务器的代码，创建了称为Apache的Web服务器软件。Apache最初是从NCSA Web服务器内核代码和一大堆补丁程序中衍生出来的。经过短短的几年时间，Apache已经成为使用最广泛的Web服务器软件，在服务器市场中占绝对优势。

Apache是一个免费的软件，用户可以免费从Apache的官方网站下载。任何人都可以参加其组成部分的开发。Apache允许世界各地的人对其提供新特性。当新代码提交到Apache Group后，Apache Group对其具体内容进行审查、测试和质量检查。如果他们满意，该代码就会被集成到Apache的主要发行版本中。

Apache的其他主要特征如下。

- 支持最新的 HTTP 协议：是最先支持 HTTP1.1 的 Web 服务器之一，其与新的 HTTP 协议完全兼容，同时与 HTTP1.0、HTTP1.1 向后兼容。Apache 还为支持新协议做好了准备。
- 简单而强大的基于文件的配置：该服务器没有为管理员提供图形用户界面，提供了三个简单但是功能异常强大的配置文件。用户可以根据需要使用这三个文件随心所欲地完成自己希望的 Apache 配置。

- 支持通用网关接口（CGI）：采用mod_cgi模块支持CGI。Apache支持CGI/1.1标准，并且提供了一些扩充。
- 支持虚拟主机：是首批既支持IP虚拟主机又支持命名虚拟主机的Web服务器之一。
- 支持HTTP认证：支持基于Web的基本认证。它还有望支持基于消息摘要的认证。
- 内部集成Perl：Perl是CGI脚本编程的事实标准。Apache对Perl提供了良好的支持，通过使用其mod_perl模块，还可以将Perl的脚本装入内存。
- 集成代理服务器：用户还可以选择Apache作为代理服务器。
- 支持SSL：由于版本法和美国法律在进出口方面的限制，Apache本身不支持SSL。但是用户可以通过安装Apache的补丁程序集合（Apache－SSL）使得Apache支持SSL。
- 支持HTTP Cookie：通过支持Cookie，可以对用户浏览Web站点进行跟踪。

### 7.2.3 安装Apache

下载并安装Apache的最新版本，是构建安全Web服务的开始，因为最新的版本总是在原有老版本的基础上添加并完善了部分服务器性能和服务器安全的功能。本节将详细介绍Apache的获取和安装过程。

Apache的RPM软件包可以在Red Hat Enterprise Linux 5 安装光盘上找到。在Red Hat Enterprise Linux 5 光盘中，其文件名为httpd-2.2.3-7.el5.rpm，执行以下命令即可完成Apache的安装：

```
#rpm -ivh httpd-2.2.3-7.el5.rpm
```

ApacheRPM将把文件安装在以下的目录中。

- ./etc/httpd/conf：该目录包含Apache的所有配置文件，包括access.conf、httpd.conf和srm.conf。参阅本章后面关于配置文件小节。
- ./etc/rc.d/：位于该目录下的目录树包含系统的启动脚本。ApacheRPM在这里安装了Web服务器的整套脚本，这些脚本可用来从命令行启动或停止服务器，并且也可在工作站关闭、启动或重新引导时自动启动或停止服务器。
- ./home/httpd：RPM在该目录安装默认的服务器图标、CGI脚本和HTML文件。如果想在其他地方保存Web内容，通过在服务器的配置文件适当的地方进行更改可以实现。
- ./usr/doc和/usr/man：RPM包含手册页和readme文件，它们被放在这些目录中。像大多数RPM软件包一样，readme文件以及其他相关的文档放在/usr/doc下的一个以服务器软件包的版本命名的目录中。
- ./usr/sbin：可执行程序放在该目录中。包括服务器程序本身，还有各种工具，如用于创建验证口令文件的htpasswd程序。
- ./var/log/http：服务器日志文件存放于该目录。在默认情况下，有两个日志文件——access_log和error_log，但是可以定义任意多个包含各种信息的自定义日志文件。

## 7.3 安全配置Apache服务器

在安装完Apache服务器后，需要对其进行正确、安全的配置。一般来说，Apache服务器有如下默认的重要配置信息。

- 主配置文件：一般位于/etc/httpd/conf/httpd.conf。

- 根文档目录：一般位于/var/www/html。
- 访问日志文件：一般位于/var/log/httpd/access_log。
- 错误日志文件：一般位于/var/log/httpd/error_log。
- Apache 模块：存放路径/usr/lib/httpd/modules。

httpd.conf文件是Apache的主配置文件，其中包含大量的Apache的配置选项，而这些选项设置的正确与否都在很大程度上关系到Apache服务器的安全和性能，因此，用户必须对它们有全面和深入的认识和掌握。其中比较常用的配置选项如表 7-2 所示。

表 7-2　httpd.conf 文件中的主要配置参数

| 配置选项 | 说　明 |
|---|---|
| ServerType | 服务器的两种类型为 standalone 和 inetd |
| ServerRoot | 设置服务器目录的绝对路径 |
| Port | 指定服务器运行在哪个端口 |
| User 和 Group | 设置用户 ID 和组 ID |
| ServerAdmin | 设置为管理服务器的 Web 管理人员的地址 |
| ServerName | 设置服务器将返回的主机名 |
| DocumentRoot | 设置为文档目录树的绝对路径 |
| UserDir | 定义和本地用户的主目录相对的目录 |
| DirectoryIndex | 指明作为目录索引的文件名 |
| TimeOut | 设置超时时间 |
| MaxSpareServers | 设置 Apache 的最大空闲进程数 |
| StartServers | 指明启动 Apache 后等待接受请求的空闲子进程数量 |
| MaxKeepAliveRequests | 设置每个连接的最大请求数 |
| KeepAlive 和 KeepAliveTimeout | 设置 session 的持续时间 |
| HostnameLookups | 设置 Apache 对客户端进行域名验证 |
| BindAddress | 设置 Apache 只监听在特定的 IP 地址 |
| LimitRequestBody | 设置 HTTP 请求的消息主体的大小 |
| LimitRequestFields | 设置 HTTP 请求的请求头的数目 |
| LimitRequestFieldSize | 设置 HTTP 请求的请求头的大小 |
| LimitRequestLine | 设置 HTTP 请求的请求行的大小 |
| MaxClients | 设置 Apache 的最大连接数 |
| LogLevel warn | 设置日志记录的级别 |

httpd.conf配置文件主要由全局环境（Section 1: Global Environment）、主服务器配置（Section 2: 'Main' server configuration）和虚拟主机（Section 3: Virtual Hosts）三个部分组成。每部分都有相应的配置语句，该文件所有配置语句的语法为“配置参数名称 参数值”的形式，如下所示：

```
StartServers 8
MinSpareServers 5
MaxSpareServers 20
```

配置语句可以放在文件中的任何地方，但为了增强文件的可读性，最好将配置语句放在相应的部分。

httpd.conf中每行包含一条语句，行末使用反斜杠“\”可以换行，但是反斜杠与下一行中间不能有任何其他字符（包括空白）。httpd.conf的配置语句除了选项的参数值以外，所有选项指令均不区分大小写，可以在每一行前用“#”号表示注释。

在默认的httpd.conf文件中，每个配置语句和参数都有详细的解释，建议初学的用户在不熟悉配置方法的情况下，先使用Apache默认的httpd.conf文件作为模板进行修改设置，并且在修改之前先做好备份，以便做了错误的修改后能够还原。否则，无意识地删除或者修改某些选项可能造成不必要的Apache服务器的安全隐患。

下面详细介绍httpd.conf文件中常用的配置参数。

### 1. ServerType 指令

ServerType指令指示服务器的类型。服务器有两种类型：standalone和xinetd。将其设置为standalone表示服务器启动一个服务进程时刻等待用户HTTP请求，当用户的请求响应后该进程并不消亡。当Server Type设置为xinetd时，对于任何传入的HTTP请求，产生一个新的服务器，该服务器在请求服务完成以后立即消亡。这可能在测试配置更改方面有用。因为每次产生一个新的服务器时都要重新装载配置文件。当然该操作非常慢，因为对于每个请求都有服务器启动的开销。

### 2. ServerRoot 指令

该指令用来设置服务器目录的绝对路径，其通知服务器到哪个位置查找所有的资源和配置文件。在配置文件中所指定的资源，有许多是相对于ServerRoot目录的。如果从RPM安装，则ServerRoot指令设置为/etc/httpd，如果从源代码安装则为/usr/local/apache。

### 3. Port 指令

该指令指定服务器运行在哪个端口上。默认为 80，这是标准的HTTP端口号。用户在某些特定情况下可能会让服务器运行在另外的端口上，例如当用户想要运行一个测试服务器而不希望其他人知道时，这时候可以指定服务器侦听非 80 端口，有些版本则用Listen指令完成该配置。修改httpd.conf里面关于Listen的选项，例如：

Listen 8000 指令就是使Apache监听 8000 端口。而如果要同时指定监听端口和监听地址，可以使用：

```
Listen 159.223.49.181:80
Listen 159.223.49.182:8000
```

这样就使得Apache同时监听在 159.223.49.181 的 80 端口和 159.223.49.182 的 8000 端口。

值得注意的是：虽然可以使用Listen指令为Apache服务器任意指定监听端口。然而，建议用户不要随便指定 1024 以上的端口，因为现在许多企业防火墙对 1024 以上的端口都不开放（为了防止木马、非常程序等），所以为了避免Apache服务也被错误“封杀”，建议用户在设置前咨询网络管理员，确定好端口。

### 4. User 和 Group 指令

这两个指令用来设置用户ID和组ID，服务器将使用它们来处理请求。通常保留这两个设置的

默认值：nobody和nogroup，并且分别在对应的/etc/passwd和/etc/group文件中验证它们。如果想使用其他的UID或GID，可以对默认设置进行修改，但是要知道，服务器将以在这里定义的用户和组的权限开始运行。这表明，假如有一个安全性的漏洞，不管是在服务器上，还是在自己的CGI程序中，这些程序都将以指定的UID运行。如果服务器以root或其他一些具有特权的用户的身份运行，那么某些人就可以利用这些安全性的漏洞对站点做一些危险的操作。除了使用名字来指定User和Group指令外，还可以使用UID和GID编号来指定它们。如果使用编号，一定要确保所指定的编号与想要指定的用户号和组号一致，并且要在编号前面加上符号“#”。

### 5. ServerAdmin 指令

该指令设置为管理服务器的Web管理人员的地址，且应该是一个有效的E-mail地址或别名，如webmaster@yourserver.com。把这一值设置为一个有效的地址十分重要，因为当服务器出现问题时，这一地址将被返回给访问者。

### 6. ServerName 指令

该指令用来设置服务器将返回的主机名，其应该被设置为一个完全限定的域名。例如，将其设置为www.yourserver.com而不是简单的www。如果服务器通过Internet访问而不是仅仅在局域网中访问，该设置尤为重要。实际上无须设置该值，除非需要返回的不是该计算机的规范名字。如果不设置该值，服务器将自行判定这一名字并把它设置为服务器的规范名字。但是，用户若想让服务器返回比较友好、易记的地址（例如www.your.domain），就需要对该指令进行设定。但是不管怎样，ServerName应该是网络的一个真正的域名系统（DNS）的名字。如果用户正在管理自己的DNS，则记住需要为主机添加一个别名。

### 7. DocumentRoot 指令

该指令设置为文档目录树的绝对路径，该路径是Apache提供文件的顶级目录。在默认情况下，它被设置为/home/httpd/html；如果是用户自己构造代码，它设置为/usr/local/apache/htdocs。例如，如果设置DocumentRoot为/webpage/main，那么当访问http://localhost/index.html时实际就是访问/webpage/main目录下的index.html文件。

### 8. UserDir 指令

该指令定义和本地用户的主目录相对的目录，可以将公共的HTML文档放入该目录中。说是相对目录是因为每个用户有自己的HTML目录。该指令默认设置为public_html。因此，每个用户在自己的主目录下都能够创建称为public_html的目录，在该目录下的HTML文档可以通过http://servername/~username访问。这里username是特定用户的名称。

### 9. DirectoryIndex 指令

该指令指明作为目录索引的文件名，例如，当请求的URL为http://www.server.com/Directory/时，指明哪个文件作为目录的索引。通常在这里放入许多文件非常有用，这样当index.html（默认的值）找不到时，可以使用另一个文件替换。该指令最有用的应用是在目录中有一个CGI程序运行，作为默认的动作。在这种情况下，该指令类似于DirectoryIndexindex.htmlindex.cgi。

### 10. TimeOut 指令

该指令在httpd.conf文件里可设置网络超时时间，其命令格式为TimeOut n。其中n为整数，单位是秒。该指令的设置一般需要通过用户经验来设定：设定超时过短，可能使用户的体验大打折扣，影响服务质量；而设定超时过长，则会使Apache服务器维护过大的“僵死”服务队列，从而影响性能。

### 11. MaxSpareServers 指令

该指令设置Apache的最大空闲进程数。命令格式如下：

```
MaxSpareServers 30
```

上述指令表明：当空闲进程超过 30 个的时候，Apache主进程会杀掉多余的空闲进程而保持空闲进程为 30 个以节省系统资源。非常繁忙的站点调节这个参数才是必要的，但是在任何时候把这个参数调到很大都不是一个好办法。同时也可以设置类似参数MinSpareServers来限制最少空闲进程数目来加快反应速度。

### 12. StartServers 指令

该指令指明启动Apache后等待接受请求的空闲子进程数量。

### 13. MaxKeepAliveRequests 指令

该指令设置每个连接的最大请求数。用法如下：MaxKeepAliveRequests 100 这样就能保证在一个连接中，如果同时请求数达到 100 就不再响应这个连接的新请求，保证了系统资源不会被某个连接大量占用。但是在实际配置中要求尽量把这个数值调高来获得较高的系统性能。

### 14. KeepAlive 和 KeepAliveTimeout 指令

该指令设置session的持续时间，例如以下两个设置：

```
KeepAlive on
KeepAliveTimeout 15
```

这样就能限制每个session的保持时间是 15 秒。session的使用可以使得很多请求都可以通过同一个TCP连接来发送，节约了网络资源和系统资源。

### 15. HostnameLookups 指令

该指令设置Apache对客户端进行域名验证。可以设置on、off或double。如果使用on，那么只进行一次反查；如果使用double，那么进行反查之后还要进行一次正向解析，只有两次的结果互相符合才行；而off就是不进行域名验证。如果为了安全，建议使用double；为了加快访问速度，建议使用off。

### 16. BindAddress 指令

该指令设置Apache只在特定的IP地址监听，从而只响应特定IP地址的HTTP请求。

### 17. LimitRequestBody 指令

该指令设置HTTP请求的消息主体的大小，单位为字节。CGI脚本一般把表单里面的内容作为消息的主体提交给服务器处理，所以现在消息主体的大小在使用CGI的时候很有用。例如使用

CGI来上传文件时，如果有如下设置：LimitRequestBody 102400，那么上传文件超过 100KB的时候就会报错。合理设置该指令可以有效地降低遭受拒绝服务攻击的风险。

#### 18. LimitRequestFields 指令

该指令设置HTTP请求的请求头的数目。合理设置该指令同样可以有效地降低遭受拒绝服务攻击的风险。

#### 19. LimitRequestFieldSize 和 LimitRequestLine 指令

这两个指令分别用来设置HTTP请求的请求头的大小和HTTP请求的请求行的大小。合理设置这两个指令可以帮助用户有效地降低Apache服务器遭受缓冲区溢出攻击的危险。

#### 20. MaxClients 指令

该指令设置Apache的最大连接数。该指令可以根据具体的Apache服务器的服务容量来进行设定。一般情况下不宜设置得过大，可以比较好地预防拒绝服务攻击。因为将该指令的值设置得过大，巨大的客户端连接极易造成Apache服务器超负荷运行。

#### 21. LogLevel warn 指令

该指令指定记录的详细等级，有 8 个等级：debug、info、notice、warn、error、crit、alert和emerg。按从详细到简略排列。我们将会在本章的 7.8 节进行详细介绍。

## 7.4 Web 服务访问控制

### 7.4.1 访问控制常用配置指令

#### 1. 配置指令

Apache实现访问控制的配置指令包括以下三种。

（1）order指令。用于指定执行允许访问控制规则或者拒绝访问控制规则的顺序。order只能设置为Order allow,deny，或Order deny,allow，分别用来表明用户先设置允许的访问地址还是先设置禁止访问的地址。Order选项用于定义默认的访问权限与Allow和Deny语句的处理顺序。Allow和Deny语句可以针对客户机的域名或IP地址进行设置，以决定哪些客户机能够访问服务器。Order语句设置的两种值的具体含义如下。

- allow, deny：默认禁止所有客户机的访问，且 Allow 语句在 Deny 语句之前被匹配。如果某条件既匹配 Deny 语句又匹配 Allow 语句，则 Deny 语句会起作用（因为 Deny 语句覆盖了 Allow 语句）。
- deny, allow：默认允许所有客户机的访问，且 Deny 语句在 Allow 语句之前被匹配。如果某条件既匹配 Deny 语句又匹配 Allow 语句，则 Allow 语句会起作用（因为 Allow 语句覆盖了 Deny 语句）。

（2）allow指令。指明允许访问的地址或地址序列。如allow from all指令表明允许所有IP来的访问请求。

（3）deny指令。指明禁止访问的地址或地址序列。如deny from all指令表明禁止所有IP来的访问请求。

### 2. 应用实例

下面举几个简单的例子对上述order、allow和deny命令的使用进行示范。

（1）在下面的例子中，admin.org域中所有主机都允许访问网站，而其他非该域中的任何主机访问都被拒绝，因为Deny在前，Allow在后，Allow语句覆盖了Deny语句：

```
Order Deny,Allow
Deny from all
Allow from admin.org
```

（2）下面的例子中，admin.org域中所有主机，除了db.admin.org子域包含的主机被拒绝访问以外，都允许访问。而所有不在admin.org域中的主机都不允许访问，因为默认状态是拒绝对服务器的访问（Allow在前，Deny在后，Deny语句覆盖了Allow语句）：

```
Order Allow,Deny
Allow from admin.org
Deny from db.admin.org
```

## 7.4.2 使用.htaccess 文件进行访问控制

任何出现在配置文件httpd.conf中的指令都可能出现在.htaccess文件中。该文件在httpd.conf文件的AccessFileName指令中指定，用于进行针对单一目录的配置（注意：该文件也只能设置对目录的访问控制）。作为系统管理员，可以指定该文件的名字和通过该文件内容覆盖的服务器配置。当站点有多组内容提供者并希望控制这些用户对其空间的操作时该指令非常有用。

值得特别注意的是：除了可以使用.htaccess文件针对单一目录进行访问控制配置外，该文件还可以在不重新启动Apache服务器的前提下使配置生效，因而使用起来非常方便。

使用该文件进行访问控制，需要经过以下两个必要的步骤。

（1）在主配置文件httpd.conf中启用并控制对.htaccess文件的使用。

（2）在需要覆盖主配置文件的目录下（也就是需要单独设定访问控制权限的目录）生成.htaccess文件，并对其进行编辑，设置访问控制权限。

### 1. 启用并控制对.htaccess 文件的使用

启用并控制对.htaccess文件的使用，首先需要使用AccessFileName参数在主配置文件中配置以下配置语句：

```
AccessFileName .htaccess
<Files ~ "^\.htaccess">
Order allow,deny
Deny from all
</Files>
```

### 2. 在.htaccess 文件中使用指令进行控制

要限制.htaccess文件能够覆盖的内容，需要使用AllowOverride指令。该指令可以进行全局设

置或者单个目录设置。要配置可以默认使用的选项，需要使用Options指令。例如，在httpd.conf文件中，可以采用上述指令建立的对/var/www/icons目录的访问控制权限的清单，如下所示：

```
<Directory "/var/www/icons">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
```

以下为各种指令的使用介绍。

（1）AllowOverrides指令。

AllowOverrides指令指定.htaccess文件可以覆盖的选项。可以对每个目录进行设置。例如，可以对主要文档root和UserDir目录的覆盖有不同的标准。该功能对于用户目录特别有用，在这些目录中用户没有访问主服务器配置文件的权限。

AllowOverrides可以设置为All、None，或者Option、FileInfo、AuthConfig、Indexes以及Limit选项的组合。这些选项含义如下。

- Options：文件可以为该目录添加没有在 Options 指令中列出的选项。
- FileInfo：.htaccess 文件包含修改文档类型信息的指令。
- AuthConfig：.htaccess 文件可能包含验证指令。
- Limit：.htaccess 文件可能包含 allow、deny、order 指令。
- Indexes：控制目录列表方式。
- None：禁止处理.htaccess 文件。
- All：表示读取以上所有指令的内容。

（2）Options指令

Options可以为None、All，或者任何Indexes、Includes、FollowSymLinks、ExecCGI以及MultiViews的组合。MultiViews不包含在All中，必须显式指定。这些选项解释如下。

- None：该目录没有启用任何可用的选项。
- All：该目录启用了所有选项，除了 MultiViews。
- Indexes：当 Index.html 文件或者另一个 DirectoryIndex 文件不存在时，目录中的文件列表将作为 HTML 页产生，显示给用户。
- Includes：该目录允许服务器侧包含（SSI）。如果允许包含但是不允许在包含中有 exec 选项，则可以写为 IncludesNoExec。基于安全的原因，对于没有完全控制权限的目录，如 UserDir 目录，该选项是一个很好的主意。
- FollowSymLinks：允许访问符号链接到文档目录的目录。这种方法不太好，不能将整个服务器全部设置为该选项。对某个目录可以这样设置，但是在仅当有足够的理由时才这样设置。该选项是一个潜在的安全隐患，因为允许 Web 用户跳出文档目录以外，并且可能潜在地允许用户访问文件系统的分区，而这些地方是不希望其他人访问的。
- ExecCGI：即使该目录不是 ScriptAlias 的目录，也在其中允许 CGI 程序。
- MultiViews：该选项是 mod_negotiation 模块的一部分。当客户请求的文档没有找到时，服务器试图计算最适合客户请求的文档。

### 3．使用.htaccess文件实例

下面以一个简单的例子来示范如何使用.htaccess文件。

（1）在Apache服务器的文档根目录下生成一个测试目录，并创建测试文件，使用以下命令即可：

```
#cd /var/www/html
#mkdir rhel5
#cd rhel5
#touch rhel5.a
#touch rhel5.b
```

（2）修改Apache服务器的主配置文件，添加以下语句：

```
<Directory "/var/www/html/rhel5">
AllowOverride Options
</Directory>
```

（3）在生成的测试目录/var/www/html/rhel5 下生成.htaccess文件，并添加以下语句：

```
Options -Indexes
```

（4）重新启动Apache服务器即可，可以看到在未配置.htaccess文件前用户可以使用客户端浏览器浏览文件，而配置文件后用户无法浏览，如图7-4和图7-5所示。另外，值得注意的是：这里的重启Apache服务器是因为步骤（2）中对主配置文件进行了修改，而不是因为修改了.htaccess文件，因为前面提到过，.htaccess文件的配置修改并不需要重新启动Apache服务器。

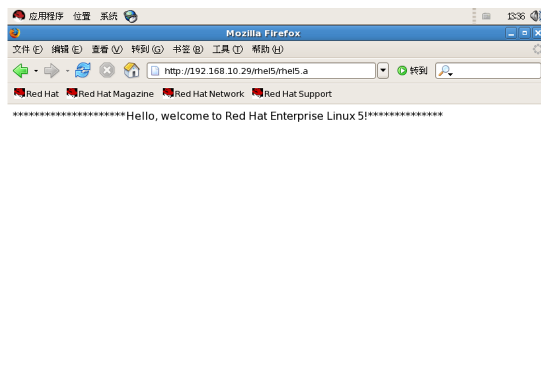



图7-4　未使用访问控制前访问目录的情况示意

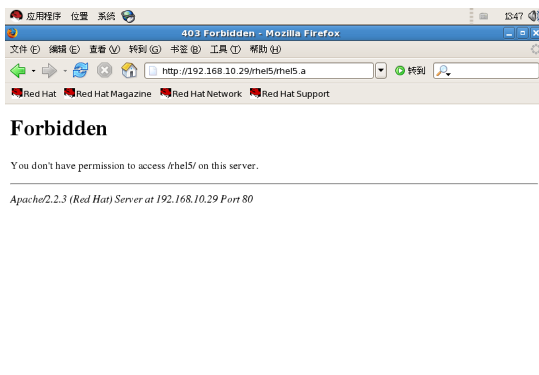



图7-5　使用访问控制后访问目录的情况示意

## 7.5　使用认证和授权保护Apache

### 7.5.1　认证和授权指令

用户认证在网络安全中是非常重要的技术之一，它是保护网络系统资源的第一道防线。用户认证控制着所有登录并检查访问用户的合法性，其目标是仅让合法用户以合法的权限访问网络系统的资源。当用户第一次访问了启用用户认证目录下的任何文件，浏览器会显示一个对话框，要

求输入正确的登录用户名和口令进行用户身份的确认。若是合法用户，则显示所访问的文件内容。此后访问该目录的每个文件时，浏览器会自动送出用户名和密码，不用再输入了，直到关闭浏览器为止。用户认证功能起到了一个屏障的作用，限制非授权用户非法访问一些私有的内容。

目前，有两种常见的认证类型：基本认证和摘要认证。

（1）基本认证（Basic）：使用最基本的用户名和密码方式进行用户认证。

（2）摘要认证（Digest）：该认证方式比基本认证要安全得多，在认证过程中额外使用了一个针对客户端的挑战（challenge）信息，可以有效地避免基本认证方式可能遇到的“重磅攻击”。值得注意的是：目前并非所有的浏览器都支持摘要认证方式。

所有的认证配置指令既可以出现在主配置文件httpd.conf中的Directory容器中，也可以出现在单独的.htaccess文件中，这个可以由用户灵活地选择使用。在认证配置过程中，需要用到以下指令选项。

- AuthName：用于定义受保护区域的名称。
- AuthType：用于指定使用的认证方式，包括上面所述的 Basic 和 Digest 两种方式。
- AuthGroupFile：用于指定认证组文件的位置。
- AuthUserFile：用户指定认证口令文件的位置。

使用上述的认证指令配置认证之后，需要为Apache服务器的访问对象，也就是指定的用户和组进行相应的授权，以便于他们对Apache服务器提供的目录和文件进行访问。为用户和组进行授权需要使用Require指令，它主要可以使用以下三种方式进行授权。

- 授权给指定的一个或者多个用户：使用 Require user 用户名 1 用户名 2…。
- 授权给指定的一个或者多个组：使用 Require group 用户名 1 用户名 2…。
- 授权给指定口令文件中的所有用户：使用 Require valid-user。

### 7.5.2 管理认证口令文件和认证组文件

要实现用户认证功能，首先要建立保存用户名和口令的文件。Apache自带的htpasswd命令提供了建立和更新存储用户名、密码的文本文件的功能。需要注意的是，这个文件必须存放在不能被网络访问的位置，以避免被下载和信息泄露。建议将口令文件存放在/etc/httpd/目录或者其子目录下。

下面的例子在/etc/httpd目录下创建一个文件名为passwd_auth的口令文件，并将用户rhel5 添加到认证口令文件。使用以下命令建立口令文件（过程中还会提示输入该用户的口令）：

```
# touch passwd_auth
# htpasswd -c /etc/httpd/passwd_auth rhel5
New password:
Re-type new password:
Adding password for user rhel5
```

命令执行的过程中系统会要求用户为rhel5 用户输入密码。上述命令中的-c选项表示无论口令文件是否已经存在，都会重新写入文件并删去原有内容。所以在添加第 2 个用户到口令文件时，就不需要使用-c选项了，如下命令所示：

```
# htpasswd /etc/httpd/passwd_auth liyang
```

### 7.5.3 认证和授权使用实例

#### 1．使用主配置文件配置用户认证及授权

在本例子中，用户可以在Apache的主配置文件httpd.conf中加入以下语句建立对目录/var/www/html/rhel5 访问的用户认证和授权机制：

```
<Directory "/var/www/html/rhel5">
AllowOverride None
AuthType Basic
AuthName "rhel5"
AuthUserFile /etc/httpd/passwd_auth
Require user rhel5 liyang
</Directory>
```

在上述例子中，使用了以下指令。

- AllowOverride：该选项定义了不使用.htaccess 文件。
- AuthType Basic：AuthType 选项定义了对用户实施认证的类型，最常用的是由 mod_auth 提供的 Basic。
- AuthName：定义了 Web 浏览器显示输入用户/密码对话框时的领域内容。
- AuthUserFile：定义了口令文件的路径，即使用 htpasswd 建立的口令文件。
- Require user：定义了允许哪些用户访问，各用户之间用空格分开。

需要注意的是：在AuthUserFile选项定义中，还需要使用以下语句事先建立认证用户patterson和liyang，该选项中的定义才能生效：

```
#htpasswd -c /etc/httpd/passwd_auth rhel5
#htpasswd /etc/httpd/passwd_auth liyang
```

#### 2．使用.htaccess 文件配置用户认证和授权

在本例子中，为了完成如上述例子同样的功能，需要先在主配置文件中加入以下语句：

```
<Directory "/var/www/html/rhel5">
AllowOverride AuthConfig
</Directory>
```

上述语句中的AllowOverride选项允许在.htaccess文件中使用认证和授权指令。

然后，在.htaccess文件中添加以下语句即可：

```
AuthType Basic
AuthName "Please Login:"
AuthUserFile /etc/httpd/passwd_auth
Require user rhel5 liyang
```

同理，在AuthUserFile选项定义中，还需要使用以下语句事先建立认证用户patterson和liyang，该选项中的定义才能生效：

```
#htpasswd -c /etc/httpd/passwd_auth rhel5
#htpasswd /etc/httpd/passwd_auth liyang
```

# 7.6 使用 Apache 中的安全模块

## 7.6.1 Apache 服务器中安全相关模块

Apache的一个优势便是其灵活的模块结构，其设计思想也是围绕模块（module）概念而展开的。安全模块是Apache Server中极其重要的组成部分。这些安全模块负责提供Apache server的访问控制和认证、授权等一系列至关重要的安全服务。

Apache有以下几类与安全相关的模块。

- mod_access 模块能够根据访问者的 IP 地址（或域名，主机名等）来控制对 Apache 服务器的访问，称之为基于主机的访问控制。
- mod_auth 模块用来控制用户和组的认证授权（Authentication）。用户名和口令存放于纯文本文件中。
- mod_auth_db 和 mod_auth_dbm 模块则分别将用户信息（如名称、组属和口令等）存放于 Berkeley-DB 及 DBM 型的小型数据库中，便于管理及提高应用效率。
- mod_auth_digest 模块则采用 MD5 数字签名的方式来进行用户的认证，但它相应的需要客户端的支持。
- mod_auth_anon 模块的功能和 mod_auth 的功能类似，只是它允许匿名登录，将用户输入的 E-mail 地址作为口令。
- mod_ssl 被 Apache 用于支持安全套接字层协议，提供 Internet 上安全交易服务，如电子商务中的一项安全措施。通过对通信字节流的加密来防止敏感信息的泄露。但是，Apache 的这种支持是建立在对 Apache 的 API 扩展来实现的，相当于一个外部模块，通过与第三方程序（如 openssl）的结合提供安全的网上交易支持。

## 7.6.2 开启安全模块

为了能够使用模块功能，模块通常以DSO（Dynamic Shared Object）的方式构建，用户应该在httpd.conf文件中使用LoadModule指令，使得能够在使用前获得模块的功能。以下为主配置文件中各个模块的情况，开启安全模块非常简单，即去掉在各安全模块所在行前的“#”符号即可，如下所示：

```
LoadModule auth_basic_module modules/mod_auth_basic.so
LoadModule auth_digest_module modules/mod_auth_digest.so
LoadModule authn_file_module modules/mod_authn_file.so
LoadModule authn_alias_module modules/mod_authn_alias.so
LoadModule authn_anon_module modules/mod_authn_anon.so
LoadModule authn_dbm_module modules/mod_authn_dbm.so
LoadModule authn_default_module modules/mod_authn_default.so
LoadModule authz_host_module modules/mod_authz_host.so
LoadModule authz_user_module modules/mod_authz_user.so
LoadModule authz_owner_module modules/mod_authz_owner.so
LoadModule authz_groupfile_module modules/mod_authz_groupfile.so
```

```
LoadModule authz_dbm_module modules/mod_authz_dbm.so
LoadModule authz_default_module modules/mod_authz_default.so
LoadModule ldap_module modules/mod_ldap.so
LoadModule authnz_ldap_module modules/mod_authnz_ldap.so
LoadModule include_module modules/mod_include.so
LoadModule log_config_module modules/mod_log_config.so
LoadModule logio_module modules/mod_logio.so
LoadModule env_module modules/mod_env.so
LoadModule ext_filter_module modules/mod_ext_filter.so
LoadModule mime_magic_module modules/mod_mime_magic.so
LoadModule expires_module modules/mod_expires.so
LoadModule deflate_module modules/mod_deflate.so
LoadModule headers_module modules/mod_headers.so
LoadModule usertrack_module modules/mod_usertrack.so
LoadModule setenvif_module modules/mod_setenvif.so
LoadModule mime_module modules/mod_mime.so
LoadModule dav_module modules/mod_dav.so
LoadModule status_module modules/mod_status.so
LoadModule autoindex_module modules/mod_autoindex.so
LoadModule info_module modules/mod_info.so
LoadModule dav_fs_module modules/mod_dav_fs.so
LoadModule vhost_alias_module modules/mod_vhost_alias.so
LoadModule negotiation_module modules/mod_negotiation.so
LoadModule dir_module modules/mod_dir.so
LoadModule actions_module modules/mod_actions.so
LoadModule speling_module modules/mod_speling.so
LoadModule userdir_module modules/mod_userdir.so
LoadModule alias_module modules/mod_alias.so
LoadModule rewrite_module modules/mod_rewrite.so
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_balancer_module modules/mod_proxy_balancer.so
LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
LoadModule cache_module modules/mod_cache.so
LoadModule suexec_module modules/mod_suexec.so
LoadModule disk_cache_module modules/mod_disk_cache.so
LoadModule file_cache_module modules/mod_file_cache.so
LoadModule mem_cache_module modules/mod_mem_cache.so
LoadModule cgi_module modules/mod_cgi.so
```

## 7.7 使用 SSL 保证 Web 通信安全

Apache服务器与客户端的通信是明文方式，很多通过HTTP协议传送数据的应用将受到黑客的威胁，信息的安全性难以得到保障。因此，本节就将对在Linux中，使用SSL技术保护Apache服务器通信作详细介绍。

### 7.7.1 SSL 简介

通常的连接方式中，通信是以非加密的形式在网络上传播的，这就有可能被非法窃听到，尤其是用于认证的口令信息。为了避免这个安全漏洞，就必须对传输过程进行加密。对HTTP传输进行加密的协议为HTTPS，它是通过SSL进行HTTP传输的协议，不但通过公用密钥的算法进行加密保证传输的安全性，而且还可以通过获得认证证书CA，保证客户连接的服务器没有被假冒。

SSL是一种国际标准的加密及身份认证通信协议，用户采用的浏览器就支持此协议。SSL（Secure Sockets Layer）最初是由美国Netscape公司研究出来的，后来成为了Internet网上安全通信与交易的标准（参见图7-6）。SSL协议使用通信双方的客户证书以及CA根证书，允许客户/服务器应用以一种不能被偷听的方式通信，在通信双方间建立起了一条安全的、可信任的通信通道。它具备以下基本特征：信息保密性、信息完整性、相互鉴定。

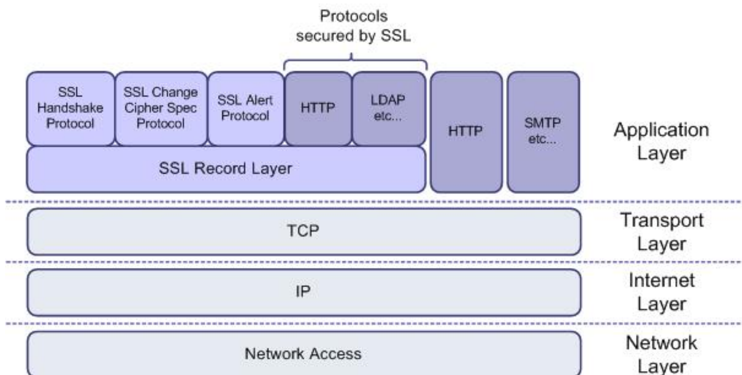



图 7-6　SSL 工作层次示意图

该协议主要使用Hash编码、加密技术，下面对这些技术进行简要介绍。

（1）Hash编码是使用Hash（散列）算法从任意长度的消息中计算Hash值的一个过程，Hash值可以说是消息的指纹，或者说是摘要。因为对于任何不同的消息，几乎总有不同的Hash值，如果有冲突的话，可以采用开链法等方法来进行避免。因此在SSL通信过程中，可以对消息的Hash值进行加密，确保传递的消息在传输过程中没有被修改。

（2）非对称加密或称之为公钥加密使用数学上相关的两个数值来对信息进行编码（加密），其中一个数字称为公钥，另一个称为私钥。公钥加密的信息可以用私钥解密，私钥加密的信息可以用公钥解密。由于公钥可以大面积发放，因此公钥加密在SSL加密通信中应用于对密钥的加密或者进行数字签名。

对称加密和非对称加密相比的区别在于对称加密中，加密信息和解密信息使用同样的密钥，因此该密钥无法公开，并且一旦知道其密码，则该加密完全失效。但是其具有加密和解密快速、方便的特点。

## 7.7.2 Apache中运用SSL的基本原理

### 1．公钥体制

加密和解密使用同一个密钥的算法，称为对称加密算法；加密和解密使用的是不同的密钥，称为非对称加密算法，公钥系统即属于非对称加密算法。对于对称加密而言，需要着重保护的是对称密钥，对于公钥算法而言，需要着重保护的是私钥。

公钥加密算法，以及衍生出的数字签名、数字证书技术，不仅广泛应用于Internet通信中，例如HTTPS协议中的SSL/TLS，在单机系统中也越来越受到重视，例如Windows XP的设备驱动程序、.NET的GAC assembly都要求数字签名。微软从Windows 98/NT4 起即提供了Cryptograph API，支持DES、RC2、RC4、IDEA等对称加密算法和RSA公钥系统等非对称加密算法，以及MD5、SHA、MAC等摘要（Digest，也称为Hash，散列）算法。

RSA公钥加密在计算机产业中被广泛使用在认证和加密。可以从RSA Data Security Inc.获得RSA公钥加密许可证。公钥加密是使用一对非对称的密码加密或解密的方法。每一对密码由公钥和私钥组成。公钥被广泛发布；私钥是隐密的，不公开。用公钥加密的数据只能够被私钥解密。反过来，使用私钥加密的数据只能用公钥解密。这个非对称的特性使得公钥加密很有用。

### 2．数字证书简介

数字证书就是互联网通信中标志通信各方身份信息的一系列数据，提供了一种在Internet上验证您身份的方式，其作用类似于司机的驾驶执照或日常生活中的身份证。它是一个由权威机构——CA机构，又称为证书授权（Certificate Authority）中心发行的，人们可以在网上用它来识别对方的身份。数字证书是一个经证书授权中心数字签名的包含公开密钥拥有者信息以及公开密钥的文件。最简单的证书包含一个公开密钥、名称以及证书授权中心的数字签名。一般情况下证书中还包括密钥的有效时间、发证机关（证书授权中心）的名称、该证书的序列号等信息，证书的格式遵循ITUT X.509 国际标准。数字证书可以应用于互联网上的电子商务活动和电子政务活动，其应用范围涉及需要身份认证及数据安全的各个行业，包括传统的商业、制造业、流通业的网上交易，以及公共事业、金融服务业、工商税务、海关、政府行政办公、教育科研单位、保险、医疗等网上作业系统。

一个标准的X.509 数字证书包含以下一些内容。

- 证书的版本信息。
- 证书的序列号，每个证书都有一个唯一的证书序列号。
- 证书所使用的签名算法。
- 证书的发行机构名称，命名规则一般采用 X.500 格式。
- 证书的有效期，现在通用的证书一般采用 UTC 时间格式，它的计时范围为 1950～2049。
- 证书所有人的名称，命名规则一般采用 X.500 格式。
- 证书所有人的公开密钥。
- 证书发行者对证书的签名。

### 3．数字证书的基本功能

基于Internet的电子商务系统技术使在网上购物的顾客能够极其方便、轻松地获得商家和企业

的信息，但同时也增加了对某些敏感或有价值的数据被滥用的风险。买方和卖方必须对于在Internet上进行的一切金融交易运作都是真实可靠的，并且要使顾客、商家和企业等交易各方都具有绝对的信心，因而因特网（Internet）电子商务系统必须保证具有十分可靠的安全保密技术，也就是说，必须保证网络安全的四大要素，即信息传输的保密性、数据交换的完整性、发送信息的不可否认性、交易者身份的确定性。

1）信息的保密性

交易中的商务信息均有保密的要求。如信用卡的账号和用户名被人知悉，就可能被盗用，订货和付款的信息被竞争对手获悉，就可能丧失商机。因此在电子商务的信息传播中一般均有加密的要求。

2）交易者身份的确定性

网上交易的双方很可能素昧平生，相隔千里。要使交易成功，首先要能确认对方的身份，对商家要考虑客户端不能是骗子，而客户也会担心网上的商店是否是一个玩弄欺诈的黑店。因此能方便而可靠地确认对方身份是交易的前提。对于为顾客或用户开展服务的银行、信用卡公司和销售商店，为了做到安全、保密、可靠地开展服务活动，都要进行身份认证的工作。对有关的销售商店来说，他们对顾客所用的信用卡的号码是不知道的，商店只能把信用卡的确认工作完全交给银行来完成。银行和信用卡公司可以采用各种保密与识别方法，确认顾客的身份是否合法，同时还要防止发生拒付款问题以及确认订货和订货收据信息等。

3）不可否认性

由于商情的千变万化，交易一旦达成是不能被否认的。否则必然会损害一方的利益。例如订购黄金，订货时金价较低，但收到订单后，金价上涨了，如收单方否认受到订单的实际时间，甚至否认收到订单的事实，则订货方就会蒙受损失。因此电子交易通信过程的各个环节都必须是不可否认的。

4）不可修改性

交易的文件是不可被修改的，如上例所举的订购黄金。供货单位在收到订单后，发现金价大幅上涨了，如其能改动文件内容，将订购数 1 吨改为 1 克，则可大幅受益，那么订货单位可能就会因此而蒙受损失。因此电子交易文件也要能做到不可修改，以保障交易的严肃和公正。

### 4．数字证书的原理

人们在感叹电子商务的巨大潜力的同时，不得不冷静地思考，在人与人互不见面的计算机互联网上进行交易和作业时，怎样才能保证交易的公正性和安全性，保证交易双方身份的真实性。国际上已经有比较成熟的安全解决方案，那就是建立安全证书体系结构。数字安全证书提供了一种在网上验证身份的方式。安全证书体制主要采用了公开密钥体制，其他还包括对称密钥加密、数字签名、数字信封等技术。

我们可以使用数字证书，通过运用对称和非对称密码体制等密码技术建立起一套严密的身份认证系统，从而保证：信息除发送方和接收方外不被其他人窃取；信息在传输过程中不被篡改；发送方能够通过数字证书来确认接收方的身份；发送方对于自己的信息不能抵赖。

数字证书采用公钥体制，即利用一对互相匹配的密钥进行加密、解密。每个用户自己设定一个特定的仅为本人所知的私有密钥（私钥），用它进行解密和签名；同时设定一个公共密钥（公钥）并由本人公开，为一组用户所共享，用于加密和验证签名。当发送一份保密文件时，发送方

使用接收方的公钥对数据加密，而接收方则使用自己的私钥解密，这样信息就可以安全无误地到达目的地了。通过数字的手段保证加密过程是一个不可逆过程，即只有用私有密钥才能解密。在公开密钥密码体制中，常用的一种是RSA体制。其数学原理是将一个大数分解成两个质数的乘积，加密和解密用的是两个不同的密钥。即使已知明文、密文和加密密钥（公开密钥），想要推导出解密密钥（私密密钥），在计算上是不可能的。按现在的计算机技术水平，要破解目前采用的1024位RSA密钥，需要上千年的计算时间。公开密钥技术解决了密钥发布的管理问题，商户可以公开其公开密钥，而保留其私有密钥。购物者可以用人人皆知的公开密钥对发送的信息进行加密，安全地传送给商户，然后由商户用自己的私有密钥进行解密。

用户也可以采用自己的私钥对信息加以处理，由于密钥仅为本人所有，这样就产生了别人无法生成的文件，也就形成了数字签名。采用数字签名，能够确认以下两点。

（1）保证信息是由签名者自己签名发送的，签名者不能否认或难以否认。

（2）保证信息自签发后到收到为止未曾作过任何修改，签发的文件是真实文件。

数字签名的具体做法如下。

（1）将报文按双方约定的HASH算法计算得到一个固定位数的报文摘要。在数学上保证：只要改动报文中任何一位，重新计算出的报文摘要值就会与原先的值不相符。这样就保证了报文的不可更改性。

（2）将该报文摘要值用发送者的私人密钥加密，然后连同原报文一起发送给接收者，而产生的报文即称数字签名。

（3）接收方收到数字签名后，用同样的Hash算法对报文计算摘要值，然后与用发送者的公开密钥进行解密解开的报文摘要值相比较。如相等则说明报文确实来自所称的发送者。

#### 5．证书授权中心

CA机构，又称为证书授权（Certificate Authority）中心，作为电子商务交易中受信任的第三方，承担公钥体系中公钥的合法性检验的责任。CA中心为每个使用公开密钥的用户发放一个数字证书，数字证书的作用是证明证书中列出的用户合法拥有证书中列出的公开密钥。CA机构的数字签名使得攻击者不能伪造和篡改证书。它负责产生、分配并管理所有参与网上交易的个体所需的数字证书，因此是安全电子交易的核心环节。

由此可见，建设证书授权（CA）中心，是开拓和规范电子商务市场必不可少的一步。为保证用户之间在网上传递信息的安全性、真实性、可靠性、完整性和不可抵赖性，不仅需要对用户的身份真实性进行验证，也需要有一个具有权威性、公正性、唯一性的机构，负责向电子商务的各个主体颁发并管理符合国内、国际安全电子交易协议标准的电子商务安全证书。

#### 6．Apache中的SSL原理

在SSL通信中，首先采用非对称加密交换信息，使得服务器获得浏览器端提供的对称加密的密钥，然后利用该密钥进行通信过程中信息的加密和解密。为了保证消息在传递过程中没有被篡改，可以加密Hash编码来确保信息的完整性。

服务器数字证书主要颁发给Web站点或其他需要安全鉴别的服务器，证明服务器的身份信息，同样客户端数字证书用于证明客户端的身份。

使用公用密钥的方式可以保证数据传输没有问题，但如果浏览器客户访问的站点被假冒，这也是一个严重的安全问题。这个问题不属于加密本身，而是要保证密钥本身的正确性问题。要保

证所获得的其他站点公用密钥为其正确的密钥，而非假冒站点的密钥，就必须通过一个认证机制，能对站点的密钥进行认证。当然即使没有经过认证，仍然可以保证信息传输的安全，只是客户不能确信访问的服务器没有被假冒。如果不是为了提供电子商务等方面对安全性要求很高的服务，一般不需要如此严格的考虑。

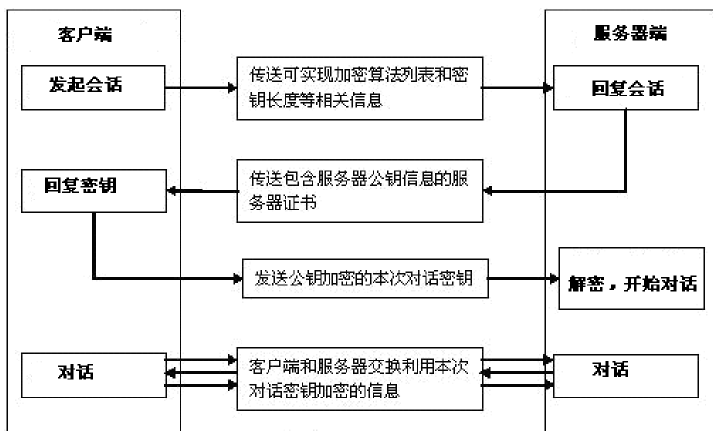



图 7-7　SSL 通信流程示意

下面给出使用SSL进行通信的过程，如图 7-7 所示。

（1）客户端向服务器端发起对话，协商传送加密算法。例如：对称加密算法有DES、RC5，密钥交换算法有RSA和DH，摘要算法有MD5 和SHA。

（2）服务器向客户端发送服务器数字证书。比如：使用DES－RSA－MD5 这对组合进行通信。客户端可以验证服务器的身份，决定是否需要建立通信。

（3）客户端向服务器传送本次对话的密钥。在检查服务器的数字证书是否正确，通过CA机构颁发的证书验证了服务器证书的真实有效性之后，客户端生成利用服务器的公钥加密的本次对话的密钥发送给服务器。

（4）服务器用自己的私钥解密获取本次通信的密钥。

（5）双方的通信正式开始。

在一般情况下，当客户端是保密信息的传递者时，他不需要数字证书验证自己身份的真实性，如用户通常使用的网上银行交易活动，客户需要将自己的隐秘信息——账号和密码发送给银行，因此银行的服务器需要安装数字证书来表明自己身份的有效性，否则将会使得信息泄露。当然，在某些安全性要求极高的 B2B（Business to Business）应用，服务器端也需要对客户端的身份进行验证，这时客户端也需要安装数字证书以保证通信时服务器可以辨别出客户端的身份，验证过程类似于服务器身份的验证过程。另外，在一些电子商务的应用中，可能还会使用到电子签名，或者为了信息交换的更加安全，会增加电子签名和消息校验码（MAC）。而在通常情况下，浏览器都会通过交互的方式来完成上述的通信过程。下面将在 Linux 中对 Apache 采用 SSL 也会做详细的介绍。

### 7.7.3　使用开源的 OpenSSL 保护 Apache 通信安全

#### 1．安装 OpenSSL

虽然Apache服务器不支持SSL，但Apache服务器有两个可以自由使用的支持SSL的相关计划，一个为Apache-SSL，它集成了Apache服务器和SSL，另一个为Apache+mod_ssl，它是通过可动态加载的模块mod_ssl来支持SSL，其中后一个是由前一个分化出的，并因为使用模块，易用性很好，

因此使用范围更为广泛。还有一些基于Apache并集成了SSL能力的商业Web服务器，使用这些商业Web服务器的主要是北美，这是因为在那里SSL使用的公开密钥的算法具备专利权，不能用于商业目的，其他的国家不必考虑这个专利问题，而可以自由地使用SSL。

Apache+mod_ssl依赖于另外一个软件：OpenSSL，它是一个可以自由使用的SSL实现，首先需要安装这个软件。用户可以从网站http://www.openssl.org/source/上下载Linux下OpenSSL的最新稳定版本：openssl-0.9.8k.tar.gz（目前发布的beta测试版本为：openssl-1.0.0-beta1.tar.gz），鉴于其稳定性，本书采用稳定版本进行安装使用。

下载源代码安装包后，使用以下的步骤安装即可。

（1）用openssl-0.9.8k.tar.gz软件包安装OpenSSL之前，首先需要对该软件包进行解压缩和解包。用以下命令完成软件包的解压缩和解包：

```
#tar xvfz openssl-0.9.8k.tar.gz
```

（2）解压缩后，进入源码的目录openssl-0.9.8k，并使用配置脚本进行环境的设置。相应的命令为：

```
//改变当前目录为 openssl-0.9.8k 目录
#cd openssl-0.9.8k

//执行该目录下配置脚本程序
#./configure
```

（3）在执行./configure之后，配置脚本会自动生成Makefile。如果在设置的过程中没有任何错误，就可以开始编译源码了。相应的命令及其显示结果如下：

```
#make & make install
```

需要说明的是：OpenSSL已经在Red Hat Enterprise Linux 5 系统中自带，其RPM包的版本为openssl-0.9.8b-7.3.el5.rpm，用户也可以在安装光盘中找到该软件包后执行以下命令进行安装：

```
#rpm -ivh openssl-0.9.8b-7.3.el5.rpm
```

安装好OpenSSL之后，就可以安装使用Apache+mod_ssl了。然而为了保证安装完全正确，需要清除原先安装的Apache服务器的其他版本，并且还要清除所有的设置文件及其默认设置文件，以避免出现安装问题。最好也删除/usr/local/www目录（或更名），以便安装程序能建立正确的初始文档目录。如果是一台没有安装过Apache服务器的新系统，便可以忽略这个步骤，而直接安装Apache+mod_ssl了。该模块也在系统Red Hat Enterprise Linux 5 中自带，其版本为mod_ssl-2.2.3-7.el5，用户可参照OpenSSL的RPM包安装方法进行安装即可。

### 2．为 OpenSSL 产生认证凭证

在采用OpenSSL进行Apache通信加密前，需要先产生与加密相关的认证凭证，如下所示：

```
# openssl genrsa -out apache.key 1024
Generating RSA private key, 1024 bit long modulus
.........................................++++++
.....................++++++
e is 65537 (0x10001)
```

```
# openssl req -new -key apache.key -out apache.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [GB]:CN
State or Province Name (full name) [Berkshire]:China
Locality Name (eg, city) [Newbury]:Beijing
Organization Name (eg, company) [My Company Ltd]:CSO
Organizational Unit Name (eg, section) []:CSO
Common Name (eg, your name or your server's hostname) []:localhost
Email Address []:CSO@ittf.org.cn

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:apacheserver
An optional company name []:apacheserver

# openssl x509 -req -days 365 -in apache.csr -signkey apache.key -out apache.crt
Signature ok
subject=/C=CN/ST=China/L=Beijing/O=CSO/OU=CSO/CN=localhost/emailAddress=CS
O@ittf.org.cn
Getting Private key
```

经过上述步骤后，将会产生三个文件：apache.csr、apache.key和apache.crt，然后把这三个文件复制到/etc/httpd/conf/ca目录下即可。

### 3．启动和关闭 OpenSSL

经过上述的步骤后，下一步就是开启具有SSL功能的Apache服务器。启动和关闭该服务器的命令如下。

- #apachectl start：启动 Apache。
- #apachectl startssl：启动 Apache ssl。
- #apachectl stop：停止 Apache。
- #apachectl restart：重新启动 Apache。
- #apachectl status：显示 Apache 的状态。
- #apachectl configtest：测试 httpd.conf 配置是否正确。

```
# /usr/local/sbin/apachectl startssl
```

此时使用start参数为仅仅启动普通Apache的httpd守护进程，而不启动其SSL，而startssl才有能启动Apache的SSL能力。如果之前Apache的守护进程正在运行，便需要使用stop参数先停止服务器运行。

然后，就可以启动Mozilla、IE或其他支持SSL的浏览器，输入URL为：https://ssl_server/来查看服务器是否有响应，https使用的默认端口为 443，如果一切正常，服务器将会返回给客户端证书，由客户端进行验证并且判断，是否接受该证书并进行下一步的通信过程。

下面以Linux下的Mozilla Firefox浏览器为例，来简要说明使用Apache+SSL服务器的过程。首先，图 7-8 给出了查看和验证该证书的相关提示；最后，图 7-9 则给出了证书验证成功后，采用SSL进行保密传输的具体界面示意。

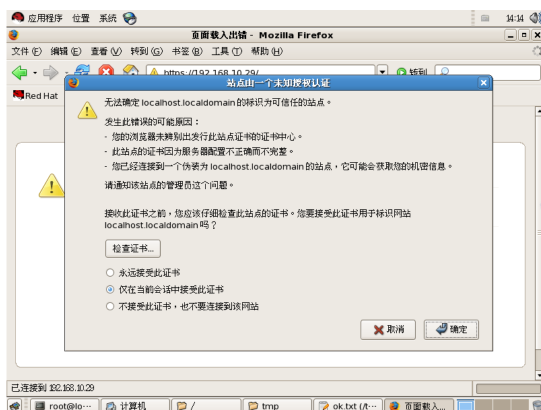



图 7-8 验证证书示意




图 7-9 证书通过验证，正常通信开始

# 7.8 Apache 日志管理和统计分析

## 7.8.1 日志管理概述

日志文件是用户管理和监控Apache安全的非常好的第一手资料，它清晰地记录了客户端访问Apache服务器资源的每一条记录，以及在访问中出现的错误信息，可以这样说，Apache可以记录Web访问中感兴趣的几乎所有信息。

当运行Apache服务器时生成以下 4 个标准的日志文件。

- 错误日志
- 访问日志
- 传输日志
- cookie 日志

其中比较常见的是访问日志（access_log）和错误日志（error_log），其中传输日志和cookie日志被Apache 2.0 以上的版本丢弃，所以本书将不讨论这两种日志。当然，如果使用SSL服务的话，还可能存在ssl_access_log、ssl_error_log和ssl_request_log三种日志文件。

另外，值得注意的是：上述几种日志文件如果长度过大，还可能生成注入access_log.1、error_log.2 等的额外文件，其格式与含义和上述几种文件相同，只不过系统自动为其进行命名而已。

### 7.8.2 与日志相关的配置指令

Apache中提供以下 4 条与日志相关的配置指令。

- ErrorLog 指令：用于指定错误日志的存放路径，使用语法为：ErrorLog 文件名。
- LogLevel：用于指定错误日志的错误登记，使用语法为：Loglevel 等级。
- LogFormat：用于为日志记录格式命名，使用语法为：LogFormat 记录格式说明字符串格式称谓。
- CustomLog：用于指定访问日志存放路径和记录格式，指定访问日志由指定的程序生成并指定日志的记录格式，使用语法为：CustomLog 日志文件名 格式称谓。

在上述几个文件当中，除了error_log和ssl_error_log之外，所有日志文件以由CustomLog和LogFormat指令指定的格式生成。这些指令在httpd.conf文件中出现。使用LogFormat指令可以定义新的日志文件格式：

```
LogFormat "%h  %l  %u  %t  \ "%> %s %b "common
```

假定使用的是common日志格式或者combined日志格式，这两种格式都在默认的配置文件中定义。表 7-3 列出了LogFormat语句可以使用的变量.

表 7-3 LogFormat 语句的变量

| 变　量 | 含　义 |
|---|---|
| %b | 发送字节，不包括 HTTP 标题 |
| %f | 文件名 |
| %{VARIABLE}e | 环境变量 VARIABLE 的内容 |
| %h | 远程主机 |
| %a | 远程 IP 地址 |
| %{HEADER}i | HEADER 内容；发送到服务器的请求的标题行 |
| %l | 远程登录名（如果提供该值，则从 identd 获得） |
| %{NOTE}n | 来自另一个模块的 NOTE 通知的内容 |
| %{HEADER}o | HEADER 的内容，回复中的标题行 |
| %p | 服务器服务于请求的规范端口 |
| %P | 服务于请求的子进程的 ID |
| %r | 请求的第一行 |
| %s | 状态。对于内部重定向的请求，该状态为初始请求—最后是%>s |
| %t | 时间，格式为 common 日志格式中的时间格式 |
| %{format}t | 时间，格式由 format 给出。可以是 strftime（3）格式 |
| %T | 服务请求花费的时间，以秒计 |
| %u | 来自 auth 的远程用户；如果返回的状态（%s）为 401 则可能是假的 |
| %U | 请求的 URL 路径 |
| %v | 服务于该请求的服务器的规范 ServerName |

在每个变量中，可以在前面设置一个条件，决定是否显示该变量。如果不显示，则显示“-”。

这些条件是数值返回值列表的形式。另外，还可以使用CustomLog指令指定日志文件的位置和格式。如果没有指定日志文件的绝对路径，则日志文件的位置假定为相对于ServerRoot。

下面是httpd.conf文件中指定日志文件的语句：

```
//
// The location and format of the access logfile（Common Logfile Format）.
// If you do not define any access logfiles within a <VirtualHost>
// container, they will be logged here.  Contrariwise, if you *do*
// define per-<VirtualHost> access logfiles, transactions will be
// logged therein and *not* in this file.
//
CustomLog logs/access_log common

ErrorLog logs/error_log
```

### 7.8.3 日志记录等级和分类

一般来说，Apache中的错误日志记录等级有如表 7-4 所示的八类。

表 7-4 错误日志记录的等级

| 紧 急 性 | 等 级 | 解 释 |
| --- | --- | --- |
| 1 | Emerg | 出现紧急状况使得系统不可用 |
| 2 | Alert | 需要立即引起注意的状况 |
| 3 | Crit | 危险情况的警告 |
| 4 | Error | 除上述 3 种情况之外的其他错误 |
| 5 | Warn | 警告信息 |
| 6 | Notice | 需要引起注意的情况，不如第 4 和第 5 类重要 |
| 7 | Info | 需要报告的一般消息 |
| 8 | Debug | 运行于 debug 模式的程序产生的消息 |

另外，在Apache中，将访问日志分为以下 4 类。

- 普通日志格式（Common Log Format，CLF）：大多数日志分析软件都支持这种格式，其在 LogFormat 指定中定义的昵称为 common。
- 参考日志格式（Referer Log Format）：记录客户访问站点的用户身份，其在 LogFormat 指定中定义的昵称为 referer。
- 代理日志格式（Agent Log Format）：记录请求的用户代理，其在 LogFormat 指定中定义的昵称为 agent。
- 综合日志格式（Combined Log Format）：即结合上述 3 种格式的日志信息，其在 LogFormat 指定中定义的昵称为 combined。

在实际的使用过程中，由于综合日志格式有效地结合了其他 3 种日志的格式和信息，所以在配制访问日志时，可以有两种方式。

（1）分别使用 3 个文件进行分别记录，相应的配置示例如下：

```
LogFormat "%h %l %u %t \ "%r\" %>s %b" common
LogFormat "%{Referer}i->%U" referer
LogFormat "%{Apache User-agent}I" agent
CustomLog logs/access_log common
CustomLog logs/referer_log referer
CustomLog logs/agent_log agent
```

（2）使用一个综合文件进行记录，相应的配置示例如下：

```
LogFormat "%h %l %u %t \ "%r\" %>s %b \"%{Referer}i\" \ "%{Apache User
-Agent}i\""combined
CustomLog logs/access_log combined
```

### 7.8.4 使用 Webalizer 对 Apache 进行日志统计和分析

在Linux下有许多流量分析软件。Webalizer就是其中一个高效、简单易用、免费的Web服务器日志及流量分析程序。其分析结果以HTML文件格式保存，可以很方便地通过Web服务器进行浏览。Internet上的很多站点都使用Webalizer进行Web服务器日志及流量分析。本节将对使用该软件进行日志统计和分析作介绍。

#### 1. Webalizer 的特点

Webalizer具有以下一些特性。

- 用 C 语言编写，具有很高的运行效率。在主频为 200MHz 的机器上，Webalizer 每秒可以分析 10000 条记录，所以分析一个 40MB 大小的日志文件只需 15 秒。
- 其支持标准的一般日志文件格式（Common Logfile Format）；除此之外，也支持几种组合日志格式（Combined Logfile Format）的变种，从而可以统计客户情况以及客户操作系统类型。并且现在 Webalizer 已经可以支持 wu-ftpd xferlog 日志格式以及 Squid 日志文件格式了。
- 支持命令行配置以及配置文件。
- 可以支持多种语言，也可以自己进行本地化工作。
- 支持多种平台，比如 UNIX、Linux、Windows、OS/2 和 MacOS 等。

#### 2. 安装 Webalizer

从Webalizer的站点http://www.mrunix.net/webalizer/download.html下载Webalizer，当前的最新版本是Webalizer-2.21-02-src.tgz。按照如下步骤进行安装：

```
//解开源代码包
#tar xvzf Webalizer-2.21-02-src.tgz
//在生成的目录中有个 lang 目录，该目录中保存了各种语言文件，但是只有繁体中文版本，可以自己
转换成简体，或者自己重新翻译一下。
//然后进入生成的目录：
#./configure
#make --with-language=chinese
//编译成功后，会产生一个 Webalizer 可执行文件，可以将其复制到/usr/sbin/目录下
```

```
#cp Webalizer /usr/sbin/
```

这样，安装就成功了，便可以对其进行配置了。

另外，Red Hat Enterprise Linux 5 也已经自带了Webalizer的RPM安装包，可以使用以下命令进行快速安装：

```
#rpm -ivh webalizer-2.01_10-30.1.rpm
```

### 3．配置 Webalizer

用户可以通过命令行配置Webalizer，也可以通过配置文件进行配置。本节将介绍使用配置文件进行配置，因为该方法使用形式比较直观，较为普遍。而且，所有通过命令行配置的选项，都将体现在配置文件上。

Webalizer的配置文件的路径为：/etc/webalizer.conf，其有一个对应的例子文件/etc/webalizer.conf.sample。在一般情况下，该配置文件的默认选项都能满足一定的应用需要，可以直接使用。在遇到特殊情况时，可以对该配置文件进行配置。下面给出配置该文件的实际例子和步骤：

```
//使用 vi 编辑 webalizer 配置文件
#vi /etc/webalizer.squid.conf
//设置访问日志的存放路径

LogFile/var/log/httpd/access_log

//设置访问日志的格式类型

LogType clf

//设置报表输出目录从默认值

OutputDir /var/www/html/usages

//使用 crontab 命令让 webalizer 每天生成 Apache 的当日流量统计分析
#crontab -e
```

### 4．使用 Webalizer

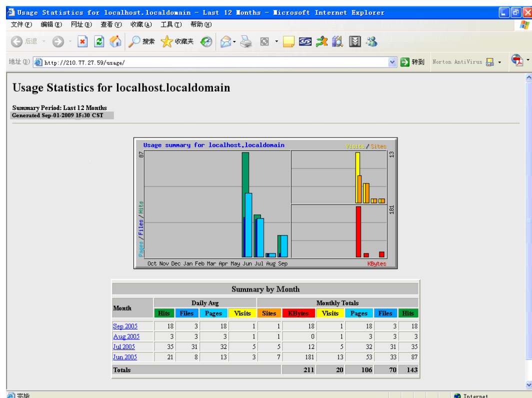
通常情况下，根据上面的讲述配置好了该软件之后，启动Apache服务器和该软件即可，如下命令所示：

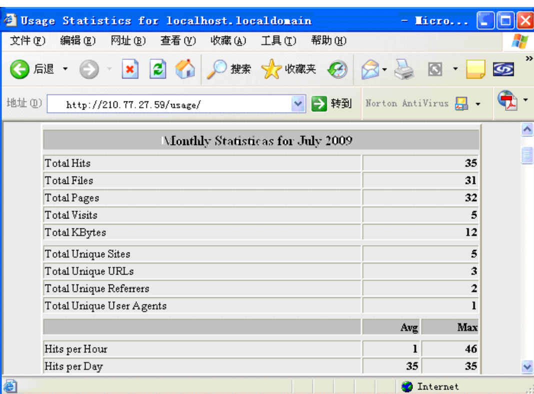
```
#service httpd start
#/usr/bin/webalizer -c /etc/webalizer.conf
```

启动Apache服务以及该软件后，就可以通过浏览器来查看Apache流量分析的结果了，在IE浏览器的地址栏内输入Apache服务器的地址：http://210.77.27.59/usage，则得到如图 7-10 所示的结果。

图 7-10 给出了流量分析的示意图以及一些数据说明，更详细的统计分析数据，可以单击图

7-10 中“Summary by Month”表格的“Jul 2009”超链接，则显示出如图 7-11 所示的详细信息。通过分析和查看这些统计图表的结果，用户可以清楚地知道Apache服务器使用的细节情况，并可以根据情况来对服务器做出适当的调整和优化。



| Monthly Statistics for July 2009 | |
|---|---|
| Total Hits | 35 |
| Total Files | 31 |
| Total Pages | 32 |
| Total Visits | 5 |
| Total KBytes | 12 |
| Total Unique Sites | 5 |
| Total Unique URLs | 3 |
| Total Unique Referrers | 2 |
| Total Unique User Agents | 1 |

| | Avg | Max |
|---|---|---|
| Hits per Hour | 1 | 46 |
| Hits per Day | 35 | 35 |

图 7-10　查看流量统计信息　　　　图 7-11　月流量详细信息报表

最后，使用Webalizer还有一个要注意的问题是：通常并不需要每个用户都有权限来查看代理服务器的流量情况，否则对于系统来说是不安全的。所以，可以使用访问控制策略来对上述分析图表的查看权限进行控制，可以在Apache的主配置文件/etc/htpd/conf/httpd.conf中进行认证和授权的配置，如下所示：

```
<Directory "/var/www/html/usage">
//使用基本认证方式
AuthType Basic
//指定认证域名称
AuthName "admin"
//指定认证口令文件存放的位置
AuthUserFile /var/www/passwd/admin
//授权给认证口令文件中的所有用户
require valid-user
</Directory>
```

认证和授权配制成功后，要创建认证口令文件以及添加用户liyang。

```
#mkdir /var/www/passwd
#cd /var/www/passwd
#htpasswd -c admin liyang
New password:
Re-type new password:
Adding passwor for user liyang
```

然后，将口令文件的属主改为Apache，并重新启动Apache。

```
#chown apache.apache admin
#service httpd restart
```

经过上面的配置后，用户在IE地址栏内输入路径：http://210.77.27.59/usage后，具有认证和授权的用户才能通过如图 7-12 所示的用户验证界面查看Apache服务器的流量分析报表信息。




图 7-12　用户登录验证界面

## 7.9 其他有效的安全措施

### 7.9.1 使用专用的用户运行 Apache 服务器

一般情况下，在Linux下启动Apache服务器的进程httpd需要root权限。由于root权限太大，存在许多潜在的对系统的安全威胁。一些管理员为了安全的原因，认为httpd服务器不可能没有安全漏洞，因而更愿意使用普通用户的权限来启动服务器。http.conf主配置文件里面有以下两个配置是Apache的安全保证，Apache在启动之后，就将其本身设置为这两个选项设置的用户和组权限进行运行，这样就降低了服务器的危险性。

```
User apache
Group apache
```

需要特别指出的是：以上两个配置在主配置文件里面是默认选项，当采用root用户身份运行httpd进程后，系统会自动将该进程的用户组和权限改为Apache，这样，httpd进程的权限就被限制在Apache用户和组范围内，因而保证了安全。

### 7.9.2 配置隐藏 Apache 服务器的版本号

Apache服务器的版本号可以作为黑客入侵的重要信息进行利用，他们通常在获得版本号后，通过网上搜索针对该版本服务器的漏洞，从而使用相应的技术和工具有针对性地入侵。因此，为了避免一些不必要的麻烦和安全隐患，可以通过主配置文件httpd.conf下的以下两个选项进行，如图 7-13 和图 7-14 所示。

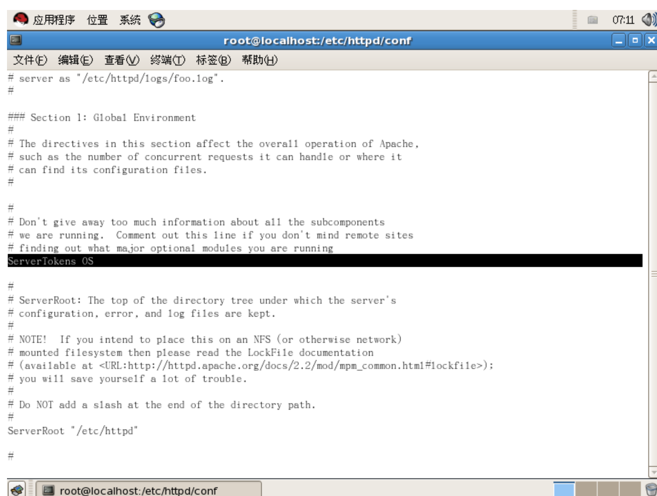



图 7-13　ServerTokens 选项示意

（1）ServerTokens。该选项用于控制服务器是否响应来自客户端的请求，向客户端输出服务器系统类型或者相应的内置模块等重要信息。Red Hat Enterprise Linux 5 操作系统在主配置文件中提供全局默认控制阈值为OS，即ServerTokens OS。它们将向客户端公开操作系统信息和相关敏感信息，所以保证安全的情况下需要在该选项后使用“ProductOnly”，即ServerTokens ProductOnly。

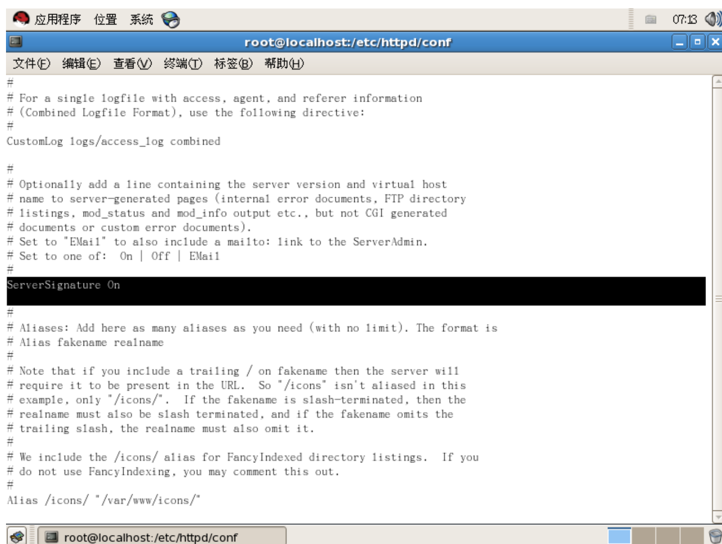



图 7-14　ServerSignature 选项示意

（2）ServerSignature。该选项控制由系统生成的页面（错误信息等）。默认情况下为off，即ServerSignature off，该情况下不输出任何页面信息。另一情况为on，即ServerSignature on，该情况下输出一行关于版本号等的相关信息。安全情况下应该将其状态设置为off。

图 7-15 和图 7-16 为安全设定这两个选项前后正常情况下和错误情况下的输出页面（通过 Rhel5 中的 Mozilla Firefox 浏览器访问 Rhel5 中的 Apache 服务器）的详细对比。可以清楚地看到，安全设定选项后，可以充分地向客户端用户隐藏 Linux 操作系统信息和 Apache 服务器版本信息。

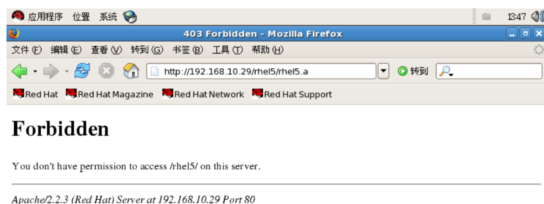



图 7-15　错误情况下未设定安全选项前示意

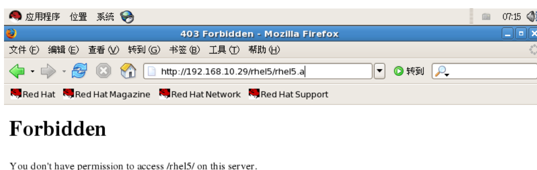



图 7-16　正常情况下使用安全设定后示意

## 7.9.3　设置虚拟目录和目录权限

要从主目录以外的其他目录中进行发布，就必须创建虚拟目录。虚拟目录是一个位于Apache的主目录外的目录，它不包含在Apache的主目录中，但在访问Web站点的用户看来，它与位于主目录中的子目录是一样的。每个虚拟目录都有一个别名，用户Web浏览器中可以通过此别名来访问虚拟目录，如http://服务器IP地址/别名/文件名，就可以访问虚拟目录下面的任何

文件了。使用虚拟目录有以下几个显著的优点。

- 便于访问：由于虚拟目录名（别名）通常要比真实目录的路径名短，因此使用虚拟目录名（别名）访问简短、方便。
- 便于移动站点中的目录：只要虚拟目录名（别名）不变，即使更改了虚拟目录的实际存放位置，无须更改目录的 URL，也不会影响用户的访问。
- 能灵活加大磁盘空间：虚拟目录能够提供的磁盘空间几乎是无限的。适合于提供对磁盘空间要求加大的 VOD 服务、个人主页服务或其他 Web 服务。
- 安全性好：由于每个虚拟目录都可以分别设置不同的访问权限，因此非常适合于不同用户对不同目录拥有不同权限的情况。此外，虚拟目录名（别名）通常只有该用户知道，其他不知道虚拟目录名的用户无法访问。黑客也不知道虚拟目录的实际存放位置，难以进行破坏。

使用Alias选项可以创建虚拟目录。在主配置文件中，Apache默认已经创建了两个虚拟目录。这两条语句分别建立了“/icons/”和“/manual”两个虚拟目录，它们对应的物理路径分别是“/var/www/icons/”和“/var/www/manual”。在主配置文件中，用户可以看到以下配置语句：

```
Alias /icons/ "/var/www/icons/"
Alias /manual "/var/www/manual"
```

在实际使用过程中，用户可以自己创建虚拟目录。比如，创建名为/user的虚拟目录，它所对应的路径为上面几个例子中常用的/var/www/html/rhel5：

```
Alias /test "/var/www/html/rhel5"
```

如果需要对其进行权限设置，可以加入以下语句：

```
<Directory "/var/www/html/rhel5">
AllowOverride None
Options Indexes
Order allow,deny
Allow from all
</Directory>
```

设置该虚拟目录和目录权限后，可以使用客户端浏览器进行测试验证，采用别名对该目录中的文件进行访问，浏览结果如图 7-17 所示。可以清楚地看到，该图和图 7-4 的显示结果一致。原因就是将/user设定成为了/var/www/html/rhel5 目录的别名。

另外，上述目录的权限设置还可以使用前面章节所介绍的.htaccess文件进行单独设置，这里不再举例赘述。

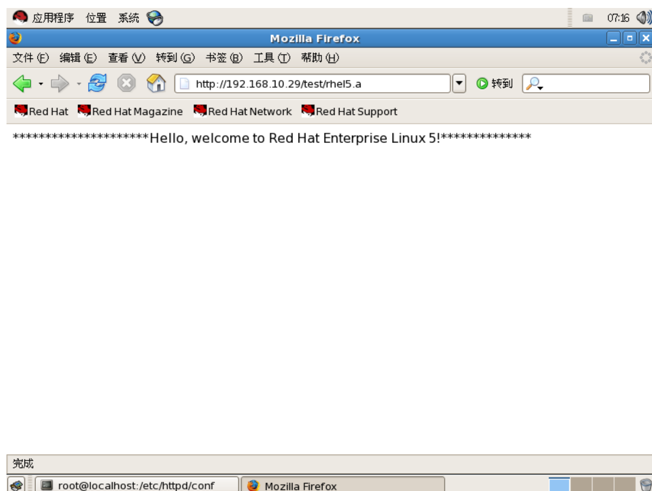



图 7-17　使用虚拟目录的测试结果

### 7.9.4 使 Web 服务运行在“监牢”中

Apache服务器需要绑定到 80 端口上来监听请求，而root是唯一有这种权限的用户，随着攻击手段和强度的增加，这样会使服务器受到相当大的威胁，一旦被利用缓冲区溢出漏洞，就可以控制整个系统。为了进一步提高系统安全性，Linux内核引入chroot机制，chroot是内核中的一个系统调用，可以通过调用函数库的chroot函数，来更改某个进程所能见到的根目录。

#### 1. “监牢”的技术特点

chroot机制即将某软件运行限制在指定目录中，保证该软件只能对该目录及其子目录的文件有所动作，从而保证整个服务器的安全。在这种情况下，即使出现黑客或者不法用户通过该软件破坏或被侵入系统，Linux系统所受的损坏也仅限于该设定的根目录，而不会影响到整个系统的其他部分。

将软件chroot化的一个问题是该软件运行时需要的所有程序、配置文件和库文件都必须事先安装到chroot目录中，通常称这个目录为chroot“监牢”。如果在“监牢”中运行httpd，那么用户根本看不到Linux文件系统中那个真正的目录，从而保证了Linux系统的安全。

在使用该技术的时候，一般情况下需要事先创建目录，并将守护进程的可执行文件httpd复制到其中。同时，由于httpd需要几个库文件，所以需要把httpd程序依赖的几个lib文件也同时拷贝到同一个目录下，因此手工完成这一工作是非常麻烦的。幸运的是，用户可以通过使用开源的Jail软件包来帮助简化chroot“监牢”建立的过程。

#### 2. 安装 Jail 软件

Jail官方网站是：http://www.jmcresearch.com/，其最新版本：jail_1.9a.tar.gz。首先从链接http://www.jmcresearch.com/static/dwn/projects/jail/jail_1.9a.tar.gz将其下载，然后执行以下命令进行源代码包的编译和安装：

```
#tar xzvf jail_1.9a.tar.gz
#cd jail/src
#make
```

#### 3. 使用 Jail 创建 chroot 监牢

Jail软件包提供了几个Perl脚本作为其核心命令，包括mkjailenv、addjailuser和addjailsw，它们位于解压后的目录jail/bin中。这几个命令的基本用途如下。

- mkjailenv：用于创建 chroot“监牢”目录，并且从真实文件系统中拷贝基本的软件环境。
- addjailsw：用于从真实文件系统中拷贝二进制可执行文件及其相关的其他文件（包括库文件、辅助性文件和设备文件）到该“监牢”中。
- addjailuser：创建新的 chroot“监牢”用户。

采用Jail创建监牢的步骤如下。

（1）首先需要停止目前运行的httpd服务，然后建立chroot目录，命令如下。该命令将chroot目录建立在路径/root/chroot/httpd下：

```
# service httpd stop
# mkjailenv /root/chroot/httpd
```

```
kjailenv
A component of Jail (version 1.9 for linux)
http://www.gsyc.inf.uc3m.es/~assman/jail/
Juan M. Casillas <assman@gsyc.inf.uc3m.es>

Making chrooted environment into /root/chroot/httpd
       Doing preinstall()
       Doing special_devices()
       Doing gen_template_password()
       Doing postinstall()
Done.
```

（2）为“监牢”添加httpd程序，命令如下：

```
# ./addjailsw  /root/chroot/httpd/ -P /usr/sbin/httpd

addjailsw
A component of Jail (version 1.9 for linux)
http://www.gsyc.inf.uc3m.es/~assman/jail/
Juan M. Casillas <assman@gsyc.inf.uc3m.es>

Guessing /usr/sbin/httpd args(0)
Warning: can't create /proc/mounts from the /proc filesystem

Done.
```

在上述过程中，用户不需要在意那些警告信息，因为Jail会调用ldd检查httpd用到的库文件。而几乎所有基于共享库的二进制可执行文件都需要上述的几个库文件。

（3）将相关文件拷贝到“监牢”的相关目录中，命令如下：

```
# mkdir -p /root/chroot/httpd/etc
# cp -a /etc/httpd /root/chroot/httpd/etc/
。。。。。。
```

添加后的目录结构如下：

```
# ll
总计 56
drwxr-xr-x 2 root root 4096 03-23 13:44 dev
drwxr-xr-x 3 root root 4096 03-23 13:46 etc
drwxr-xr-x 2 root root 4096 03-23 13:46 lib
drwxr-xr-x 2 root root 4096 03-23 13:46 selinux
drwsrwxrwx 2 root root 4096 03-23 13:46 tmp
drwxr-xr-x 4 root root 4096 03-23 13:46 usr
drwxr-xr-x 3 root root 4096 03-23 13:46 var
```

（4）重新启动httpd，并使用ps命令检查httpd进程，发现该进程已经运行在“监牢”中，如下所示：

```
 # ps -aux | grep httpd
 Warning: bad syntax, perhaps a bogus '-'? See /usr/share/doc/procps-3.2.7/FAQ
 root        3546   0.6   0.3    3828   1712  pts/2      S       13:57    0:00
/usr/sbin/nss_pcache off /etc/httpd/alias
 root        3550  14.2   3.6   49388  17788  ?            Rsl   13:57    0:00
/root/chroot/httpd/httpd
 apache      3559   0.2   1.4   49388   6888  ?           S       13:57    0:00
/root/chroot/httpd/httpd
 apache      3560   0.2   1.4   49388   6888  ?           S       13:57    0:00
/root/chroot/httpd/httpd
 apache      3561   0.2   1.4   49388   6888  ?           S       13:57    0:00
/root/chroot/httpd/httpd
 apache      3562   0.2   1.4   49388   6888  ?           S       13:57    0:00
/root/chroot/httpd/httpd
 apache      3563   0.2   1.4   49388   6888  ?           S       13:57    0:00
/root/chroot/httpd/httpd
 apache      3564   0.2   1.4   49388   6888  ?           S       13:57    0:00
/root/chroot/httpd/httpd
 apache      3565   0.2   1.4   49388   6888  ?           S       13:57    0:00
/root/chroot/httpd/httpd
 apache      3566   0.2   1.4   49388   6888  ?           S       13:57    0:00
/root/chroot/httpd/httpd
 root       3568  0.0  0.1   4124   668 pts/2    R+   13:57   0:00 grep httpd
```

## 7.10 Web 系统安全架构防护要点

### 7.10.1 Web 系统风险分析

从风险发生的位置来看，企业Web网站系统主要面临以下两类安全风险。

#### 1．用户侧和传输网络侧

用户侧和传输网络侧面临的风险如下。

（1）恶意用户采用黑客工具构造恶意报文对暴露在公网的网上系统进行拒绝服务攻击。

（2）恶意用户通过Web浏览器的登录界面对合法用户的用户名和密码进行猜测，从而冒充合法用户进行网页访问和系统使用。

（3）恶意用户通过构造非法的、可能被网上系统错误识别和执行的代码嵌入在提交的表单中，引起不正常的信息泄露，甚至系统崩溃。

（4）恶意用户可能在传输网络中通过非法窃取合法用户的通信报文，从而获得本不应该获得的敏感信息。

（5）用户被引导进入其他的非法网站，如现在流行的钓鱼网站（phishing）等等，从而在不知情的情况下泄露个人机密信息，造成经济损失。

……

#### 2. 系统服务器侧

系统服务器侧面临的风险如下。

（1）面临来自用户侧的拒绝服务攻击、分布式拒绝服务攻击。

（2）面临在遭受攻击后，由于服务器侧网络架构划分和隔离措施不严谨，可能造成“雪崩效应”，整个服务器机群的瘫痪，比如，由于Web服务器瘫痪，导致后台数据库服务器、管理服务器等的连锁瘫痪。

（3）面临由于服务器侧的软件实现不合理，尤其是数据库权限和视图等的不合理，导致用户的错误或者非法输入引起机密信息泄露或者是遭受SQL injection攻击等。

（4）面临在服务器遭受攻击后，没有相应的备份服务器接管服务，造成服务终端，引起用户业务体验严重受损。

（5）面临在服务器遭受攻击后，没有健全的灾难备份和恢复措施对关键的业务数据及其业务进行恢复。

### 7.10.2　方案的原则和思路

依据网络安全领域最为流行的4A（认证Authentication、账号Account、授权Authorization、审计Audit）、P2DR模型（策略Policy、保护Protection、监测Detection、反应Response）及ISO等主流标准的要求，按Web系统的应用需求，切实保证运维的网站能够从根本上解决黑客攻击防御和事后的审计和追踪问题，并特别强调网站的安全设计和防护原则从天生上就有应对黑客攻击的“免疫力”，所以从五大方面进行考虑。

（1）网络拓扑结构：合理的拓扑结构能够有效地抑制黑客攻击，即便在黑客攻击后也能使得攻击的影响降低到最小程度。

（2）安全原则：用户认证、账号管理、加密传输等原则的综合部署和使用，能从根本上多层面地增强网站的健壮性、数据的完整性和可靠性，从而保证网络和信息安全。

（3）审计和追踪：强大和适时的审计和追踪，能够使得网站的防御体系尽快地从黑客攻击中解脱出来，完成对黑客的追踪，并通过相应手段来增强网站的抗攻击性。

（4）备份和灾难恢复：能够保证在黑客攻击以及自然灾害下，网站系统运维的365*24*7（小时）不间断地业务运行。

（5）自我漏洞挖掘及防护：能够周期性、自发地对Web系统的漏洞进行自我挖掘，并根据挖掘的漏洞通过各种安全机制和补丁等方式进行防护，以有效地避免“零日攻击”等。

根据上述原则，建议从以下九个方面进行方案制定。

#### 1. Web系统用户的身份认证和鉴权机制

（1）采用用户名+密码验证，确认用户登录身份，并根据数据库中预设的权限，向用户展示相应的视图和表单。

（2）对于重要的Web系统应用，需要根据PKI机制，验证用户提供的证书，从而对用户身份认证（服务器对客户端认证），并确保交易的不可抵赖性。证书的提供可以采用以下两种方式。

- 文件证书：保存在用户磁盘和文件系统上，有一定的安全风险，但是可以免费。
- USB 设备存储的证书：保存在 USB 设备上，安全性很高，但是目前使用一般需要对用户收费。

### 2．Web 系统数据的加密传输和用户对 Web 系统服务器的验证

对于使用Web浏览器的网上系统应用，采用SSL+数字证书结合的方式（即HTTPS协议），保证通信数据的加密传输，同时也保证了用户端对服务器端的认证，避免用户被冒充合法网站的“钓鱼网站”欺骗，从而泄露机密信息（用户名和密码等），造成不可挽回的经济损失。

### 3．基于用户账号使用行为的日志记录及其审计

系统服务器侧应根据账号，对用户的使用行为进行详细的日志记录和审计，通过上述因素的日志记录，进行阶段性的审计（时间间隔应该比较小），从而做到发现用户账号的盗用、恶意使用等问题，尽早进行处理。

### 4．恶意用户流量的检测、过滤及阻断

系统服务器侧应部署IDS入侵检测系统、IPS入侵防护系统、防火墙等设备，或者部署目前高效、流行的UTM（统一威胁管理）设备，对恶意用户采用的各种攻击手段进行检测和防护，重点过滤恶意流量、突发流量等。

### 5．对用户的非正常应用请求的过滤和处理

系统的服务器端，尤其是数据库服务器端，应该通过配置和增加对用户非正常应用请求的过滤和处理模块，以避免由于数据库的自身漏洞未及时打上补丁，而遭受目前流行的SQL 注入攻击等。

### 6．Web 系统服务器侧的合理子网划分及流量分割

系统服务器侧包括大量的服务器类型，包括数据库服务器、Web服务器、FTP服务器、邮件服务器等，为了避免由于恶意流量造成的某种服务器崩溃，而引起的攻击后果扩散，并最终导致其他服务器也发生“雪崩效应”，则需要通过子网隔离（比如VLAN划分）、DMZ区域的设定等方式来将这些服务器放置在不同的安全域当中，做到流量和数据的安全隔离，从而将服务器端在遭受攻击后对整个业务系统及其他内网资源和数据造成的影响尽量控制在最低的范围内。

### 7．Web 系统服务器侧的负载均衡及负载保护机制

（1）系统面临着巨大的服务量，服务器端的设备基本上都需要有多台服务器进行业务分担，这样才能提高性能，避免处理瓶颈的出现，因此，需要采用合理的负载均衡和负载保护机制。

（2）对各服务器的业务流量进行有效地分担，可按照Round Robin、LRU等方式来进行负载均衡。

（3）负载保护机制需要实时地对每台服务器的CPU资源、内存资源等进行评估，如果一旦超过设定的阈值（80%或者以上），将马上进行过载保护，从而保证服务器自身的安全。

#### 8．Web 系统服务器侧的灾难备份及恢复策略

任何系统都不能说 100%的安全，都需要考虑在遭受攻击或者是经受自然灾害后的备份恢复工作，需要着重考虑以下几点。

（1）选择合适的备份策略，做好提前备份，包括全备份、差分备份、增量备份等。

（2）选择合适的备份介质，包括磁带、光盘、RAID磁盘阵列等。

（3）选择合适的备份地点，包括本地备份、远程备份等。

（4）选择合适的备份技术，包括NAS、SAN、DAS等。

（5）作好备份的后期维护和安全审计跟踪。

#### 9．Web 系统服务器的安全管理

系统功能复杂，业务数据敏感，保密级别比较高，并且对不同管理人员的权限、角色要求都不尽相同，为了保证安全管理，避免内部管理中出现安全问题，建议作如下要求。

（1）严格划分管理人员的角色及其对应的权限，避免一权独揽，引起安全隐患。

（2）做好服务器机房的物理条件管理，避免电磁泄漏、避免由于静电等引起的故障。

（3）应做好服务器管理员的账号/口令管理，要求使用强口令，避免内部人员盗用。

（4）做好服务器的端口最小化管理，避免内部人员扫描得出服务器的不必要的开放端口及其漏洞，实行内部攻击。

（5）做好服务器系统软件、应用软件的日志管理和补丁管理工作，便于审计和避免由于安全漏洞而遭受到内部人员的攻击。

（6）根据业务和数据的机密等级需求，严格划分服务器的安全域，避免信息泄露。

（7）网站漏洞自我挖掘及防护：采用漏洞扫描和挖掘设备，对内网各服务器进行阶段性地扫描，并根据扫描所得的风险和漏洞进行及时地修补，以实现该漏洞被黑客使用之前进行自行修复的目的。

### 7.10.3 网络拓扑及要点剖析

#### 1．网络拓扑

方案的网络拓扑如图 7-18 所示。

#### 2．部署要点解析

1）防火墙的设置

图 7-18 中所示防火墙的设置原则如下。

- 采用外部和内外防火墙双重设置，以切实保障外部流量进入 HTTP 反向代理服务器（DMZ 区域）和内部网络前均通过安全检测。

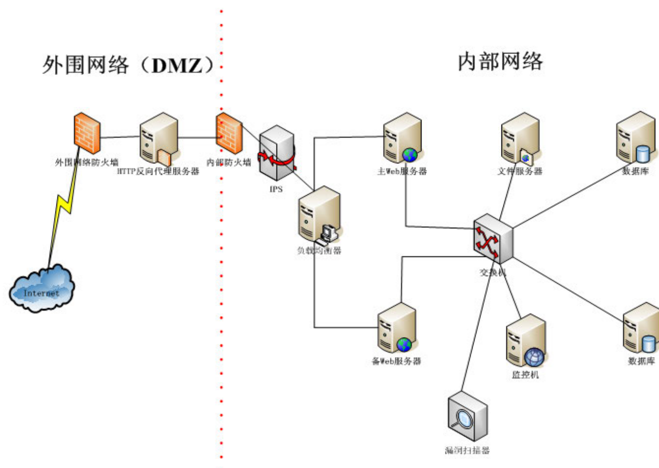



图 7-18　Web 网站安全防护整体拓扑图

- 内部和外部防火墙的使用应尽量采取不同厂家和不同型号的防火墙设备，以提高防火墙的防护性能。

2）HTTP反向代理服务器的设置

通过设置反向代理服务器，可以起到以下安全防护作用。

- 隐藏内部 Web 服务器的 IP 地址，外部用户根本感觉不到反向代理服务器的存在，极大地降低网络内部 Web 服务器被攻击的风险和概率。
- 反向代理服务器可以缓存内部 Web 服务器的部分数据，可以减轻内部服务器的负载压力，提升服务质量。
- 作为一个堡垒主机，即使反向代理服务器遭受攻击，由于内外部防火墙的设置，也不会影响到内部的网络服务器安全。

3）IPS的设置

使用上述方法设置IPS，可以起到以下安全防护和异常阻断作用。

- 对从反向代理服务器流向内部的流量和请求在内部的入口端进行过滤，成为防火墙后的第二层防护点，从源头保证内部安全。
- 对内网的异常状况可以进行实时的检测，即使有威胁和异常源自内部而对服务器发生破坏作用，如篡改网页、删除文件等，也能进行有效地监控、审计和记录，从而保证内网安全。

# 第8章

# 谨小慎微：企业基础网络服务防护

## 本章导读

作为500强企业来说，提供必要的企业基础网络服务是必不可少的。在FTP、NFS等网络服务不断消亡的同时，DNS服务和电子邮件服务却变得越来越重要。大型企业有众多的服务器资源和服务资源，因此建设企业内部DNS就变得非常重要；大型企业都把电子邮件服务当作仅次于Web服务的第二大重要基础性服务，那么电子邮件服务器的搭建也是必不可少的。

搭建这些基础网络服务不是一件难事，然而保证这些基础网络服务的安全运行则是一件棘手的事情。本章从分析这些基础网络服务面临的安全风险出发，给出如何对这些服务进行安全防护的方法和技术。

# 8.1 企业基础网络服务安全风险分析

## 8.1.1 企业域名服务安全风险分析

DNS系统用于命名组织到域层次结构中的计算机和网络服务。它主要应用于Internet等TCP/IP网络中，通过用户友好的名称查找计算机和服务。当用户在应用程序中输入DNS名称时，DNS服务可以将此名称解析为与之相关的其他信息。可以说，DNS服务器是非常关键的互联网基础设施。因此，保证DNS服务器及其提供的服务安全，也是Linux网络安全中的一个重要研究问题。

在系统设计方面，DNS的设计受到当时条件的限制，因而存在许多设计缺陷问题。

（1）单点故障。DNS采用层次化的树型结构，由树叶走向树根就可以形成一个全域名（Fully Qualified Domain Name，FQDN），DNS服务器作为该FQDN唯一对外的域名数据库和对内部提供递归域名查询的系统，因而其安全和稳定就存在单点故障风险。

（2）无认证机制。DNS没有提供认证机制，查询者在收到应答时无法确认应答信息的真假，就容易导致DNS欺骗。假设当提交给某个域名服务器的域名解析请求的数据包被黑客截获，然后黑客将一个虚假的IP地址作为应答信息返回给请求者，那么原始请求者就会把这个虚假的IP地址作为它所要请求的域名而进行连接，显然它被欺骗到了别处而连接不上原本想要连接的那个域名，这样就导致了DNS欺骗，如图 8-1 所示。




图 8-1　DNS 欺骗原理示意

（3）访问量和维护量巨大以及远距离集中式数据库。单个名字服务器不得不处理所有DNS查询消息，并保存所有因特网主机的记录，数据库会相当巨大，需要为每台新增的主机频繁更新，而且单台名字服务器主机不可能在所有请求查询的客户主机附近，就可能导致相当大的延迟。

（4）BIND（Berkeley Internet Name Domain）的漏洞：BIND是域名软件，它在提供高效服务的同时也存在许多的安全性漏洞。现已证明在BIND版本 4 和 8 上存在缺陷，攻击者利用这些缺陷能成功地进行DNS欺骗攻击，这些漏洞可以被利用，并取得系统最高权限。构成严重威胁的漏洞主要有两种：一种是缓冲区溢出漏洞，严重的可以使攻击者在DNS服务器上执行任意指令，如BIND SIG Cached RR Ovemow DoS（CAN.2002-1219）在BIND 4 和BIND 8 中存在一个远程缓冲溢出缺陷，该缺陷使得攻击者可以在DNS服务器上运行任意指令。另一种是DoS漏洞，受攻击后DNS服务器不能提供正常服务，而且其所辖的子网无法正常工作。

因此，在实际运行中，企业DNS面临以下一些安全威胁。

（1）内部攻击：攻击者在非法或合法地控制一台DNS服务器后，可以直接操作域名数据库，修改指定域名所对应的IP为自己所控制的主机IP，当客户发出对指定域名的查询请求后，将得到伪造的IP地址。

（2）序列号攻击：DNS协议格式中定义了用来匹配请求数据包和响应数据包序列ID，欺骗者利用序列号伪装成DNS服务器向客户端发送DNS响应数据包，在DNS服务器发送的真实DNS响应数据包之前到达客户端，从而将客户端带到攻击者所希望的网站，进行DNS欺骗。

（3）信息插入攻击：攻击者可以在DNS应答报文中随意添加某些信息，指示权威域名服务器的域名及IP，那么在被影响的域名服务器上查询该域的请求都会被转向攻击者所指定的域名服务器上去，从而威胁到网络数据的完整性。

（4）缓存（cache）中毒：DNS使用超高速缓存，即当一个名字服务器收到有关域名和IP的映射信息时，它会将该信息存放在高速缓存中。当再次遇到相同的映射请求，能直接使用缓存中的结果，这种映射表是动态更新的，刷新也是有时限的，这样假冒者如果在下次更新之前成功地修改了DNS服务器上的映射缓存，就可以进行DNS欺骗或者DoS（Denial of Service）拒绝服务攻击了。

（5）信息泄露：BIND的默认设置允许任何人进行区传送，区传送可能会造成信息泄露，区传送一般用于主服务器和副服务器之间的数据同步，副服务器可以从主服务器获取最新区数据文件的副本，也就可以获得整个授权区域内的所有主机信息。一旦这些信息泄露，攻击者就可以根据它轻松地推测主服务器的网络结构，并从这些信息中判断其功能或发现那些防范措施较弱的机器。

（6）不安全的动态更新：随着动态主机配置协议（DHCP）的出现，客户计算机由DHCP服务器动态分配IP地址，使原来手工更新其A记录和PTR记录变得很难管理。因此在RFC2136 中提出了DNS动态更新，使得DNS客户端在IP地址或名称出现更改的任何时候都可利用DNS服务器来注册和动态更新其资源记录。尽管DNS动态更新协议规定只有经过授权的主机才能动态更新服务器的zone file，但是攻击者还是可以利用IP欺骗伪装成DNS服务器信任的主机对区数据进行添加、删除和替换。

### 8.1.2　企业电子邮件服务安全风险分析

一般来说，电子邮件系统面临以下几种安全威胁。

- 电子邮件系统自身的安全问题：电子邮件系统自身作为一个网络服务器，存在着配置和误操作上的安全威胁和隐患，如没有合理配置服务器的相关配置文件中的重要选项等，极有可能造成潜在的安全隐患。另外，电子邮件系统版本的及时更新与否也影响到其安全。
- 垃圾邮件问题：垃圾邮件问题是当今最让网络用户头疼的顽疾之一。许多不请自来的垃圾邮件不但占据网络带宽，也极大地消耗了邮件服务器的存储资源，给用户带来非常大的不便。如何应对该问题，是电子邮件系统面临的最大挑战。
- 开放性中继的安全问题：如果设置不合理，将直接引起电子邮件系统的滥用，甚至会成为垃圾邮件的温床，它可以说是电子邮件系统中的“定时炸弹”。

## 8.2　企业域名服务安全防护

### 8.2.1　正确配置 DNS 相关文件

在使用DNS服务器之前，需要对与之相关的配置文件进行安全配置，因而首先需要了解这些基本文件，表 8-1 详细给出了几种主要的与DNS有关的文件以及详细描述。

表 8-1　DNS 相关配置文件介绍

| 文件名称 | 说　明 |
| --- | --- |
| /etc/name.conf | 它是 DNS 服务器的主文件，通过它可以设置一般的 name 参数，指向该服务器使用的域数据库的信息源 |
| /var/named/named.ca | 包含了 Internet 根服务器名字和地址，接到客户端查询请求时，如果 Cache 中找不到，就通过根服务器逐级查询 |
| /var/named/localhost.zone | localhost 区文件，用于将名字 localhost 转换为本地回送 IP 地址（127.0.0.1） |
| /var/named/name.local | localhost 区反向域名解析文件，用于将本地 IP 地址（127.0.0.1）转化为回送方 localhost 名字 |
| /var/named/name2ip.conf | 用户配置的正向解析文件，将主机名映射为 IP 地址 |
| /var/named/ip2name.conf | 用户配置的反向解析文件，将 IP 地址映射为主机名 |

### 1．named.conf 主配置文件

在使用named.conf进行配置时，需要了解以下常用的配置语句，如表 8-2 所示。

表 8-2　named.conf 主配置文件配置语句说明

| 配置语句 | 说　明 |
| --- | --- |
| zone | 定义一个区 |
| options | 定义全局配置选项 |
| include | 将其他文件包含到本配置文件中使用 |
| controls | 定义 rndc 命令使用的控制通道 |
| acl | 定义基于 IP 地址的访问控制 |
| Key | 定义授权的安全密钥 |

根据在实际应用中的广泛程度和重要性，下面我们着重对option语句和zone声明的使用进行介绍。

1）使用option语句

option语句的使用语法如下：

```
option {
  配置子句 1;
  配置子句 2;
};
```

在上述语法中，其配置子句常用的主要有以下两类。

- directory：该子句后接目录路径，主要用于定义服务器区配置文件的工作目录，如/home 等。
- forwarders：该子句后接 IP 地址，定义转发器。

2）使用zone声明

区声明是主配置文件中常用而且最重要的部分，它一般有说明域名、服务器类型以及域信息源三个重要部分。它的语法为：

```
  zone "zone_name" IN {
```

```
        type 子句;
        file  子句;
        其他子句;
    };
```

那么，围绕上述三个重要部分，区声明语句有以下两类子句。

- type：其主要有三种，master（说明一个区为主域名服务器）、slave（说明一个区为辅助域名服务器）和hint（说明一个区为启动时初始化高速缓存的域名服务器）。
- file：后接文件路径，主要说明一个区的域信息源的路径。

3）使用ACL（访问控制列表）

访问控制列表（ACL，Access Control List）是一个被命名的地址匹配列表。使用访问控制列表可以使配置简单而清晰，一次定义之后可以在多处使用，不会使配置文件因为大量的IP地址而变得混乱。

要定义访问控制列表，可以在BIND的主配置文件/etc/named.conf中使用ACL语句来实现。ACL语句的语法为：

```
acl  acl_name {
  address_match_list;
};
```

BIND里默认预定义了4个名称的地址匹配列表，它们可以直接使用，分别如下。

- Any：表示所有主机。
- Localhost：表示本机。
- Localnets：表示本地网络上的所有主机。
- None：表示不匹配任何主机。

需要注意的是：ACL是named.conf中的顶级语句，不能将其嵌入其他的语句。要使用用户自己定义的访问控制列表，必须在使用之前定义。因为可以在options语句里使用访问控制列表，所以定义访问控制列表的ACL语句应该位于options语句之前。

另外，为了便于维护管理员定义的访问控制列表，可以将所有定义ACL的语句存放在单独的文件/etc/named.conf.acls中，然后在主配置文件/etc /named.conf中添加如下语句：

```
include "/etc/named.conf.options";
```

之前添加如下的配置行：

```
include "/etc/named.conf.acls";
```

定义了ACL之后，可以在以下的子句中使用。

- allow-query options,zone：指定哪些主机或网络可以查询本服务器或区，默认的是允许所有主机进行查询。
- allow-transfer options,zone：指定哪些主机允许和本地服务器进行域传输，默认值是允许和所有主机进行域传输。
- allow-recursion options：指定哪些主机可以进行递归查询。如果没有设定，默认是允许所有主机进行递归查询的。注意禁止一台主机的递归查询，并不能阻止这台主机查询已经存在于服务器缓存中的数据。
- allow-update zone：指定哪些主机允许为主域名服务器提交动态DNS更新。默认为拒绝

任何主机进行更新。

- blackhole options：指定不接收来自哪些主机的查询请求和地址解析。默认值是 none。

上面列出的一些配置子句既可以出现在全局配置options语句里，又可以出现在zone声明语句里，当在两处同时出现时，zone声明语句中的配置将会覆盖全局配置options语句中的配置。

### 2．区文件

区文件定义了一个区的域名信息，通常也称域名数据库文件。每个区文件都是由若干个资源记录（Resource Records，RR）和区文件指令所组成。

1）资源记录

每个区域文件都是由SOA RR开始，同时包括NS RR。对于正向解析文件还包括A RR、MX RR、CNAME RR等等；而对于反向解析文件还包括PTR RR。

RR具有基本的格式。标准资源记录的基本格式是：

```
[name]      [ttl]     IN     type     redata
```

各个字节之间由空格或制表符分隔。表 8-3 描述了这些字段的含义。

表 8-3　标准资源记录中的字段

<table>
<tr><th>字　段</th><th colspan="2">说　明</th></tr>
<tr><td rowspan="6">name</td><td colspan="2">资源记录引用的域对象名，可以是一台单独的主机，也可以是整个域</td></tr>
<tr><td>取值</td><td>说明</td></tr>
<tr><td>•</td><td>根域</td></tr>
<tr><td>@</td><td>默认域，可以在文件中使用$ORIGIN domain 来说明默认域</td></tr>
<tr><td>标准域名</td><td>或是以“.”结束的域名，或是一个相对域名</td></tr>
<tr><td>空</td><td>该记录适用于最后一个带有名字的域对象</td></tr>
<tr><td>ttl(time to live)</td><td colspan="2">寿命字段。它以秒为单位定义该资源记录中的信息存放在高速缓存中的时间长度。通常该字段值为空，表示采用 SOA 中的最小值 ttl</td></tr>
<tr><td>IN</td><td colspan="2">将该记录标识为一个 Internet DNS 资源记录</td></tr>
<tr><td rowspan="10">type</td><td colspan="2">标识者是哪一类资源记录</td></tr>
<tr><td>记录类型</td><td>功能说明</td></tr>
<tr><td>A（address）</td><td>用于将主机名转换为 IP 地址，任何一个主机只能有一个 A 记录</td></tr>
<tr><td>CNAME（Canonical NAME）</td><td>给定主机的别名，主机的规范名在 A 记录中给出</td></tr>
<tr><td>HINFO（Host INFOrmation）</td><td>描述主机的信息</td></tr>
<tr><td>MX（Mail eXchanger）</td><td>邮件交换记录。告诉邮件进程把邮件发送到另一个系统。此系统值知道如何将邮件传送到它的最后总目的地</td></tr>
<tr><td>NS（Nnme Server）</td><td>标识一个域的域名服务器</td></tr>
<tr><td>PRT（domain name PoinTeR）</td><td>将地址转换为主机名</td></tr>
<tr><td>SOA（Start Of Authority）</td><td>SOA 记录表示一个授权区的开始。SOA 记录后的所有信息是控制这个域的。每个配置文件都必须包含一个 SOA 记录，以标识服务器所管理的起始地方。配置文件的第一个记录必须是 SOA 记录</td></tr>
</table>

续表

| 字　段 | 说　明 | | | |
|---|---|---|---|---|
| Rdata | 指定与这个资源记录有关的数据，数据字段的内容取决于类型字段 | | | |
| | 记录类型 | 数据 | | 说明 |
| | A | IP address | | IP 地址 |
| | CNAME | Canonical-name | | 别名 |
| | HINFO | Hardware | | “机器硬件名” |
| | | Os-type | | 操作系统名 |
| | MX | Preference-value | | 优先级别数字（数字的值越小级别越高） |
| | | Mailer-exchanger | | 邮件服务器名字 |
| | NS | Name-server | | 域名服务器的名字 |
| | PTR | Real-name | | 主机的真实域名 |
| | SOA | Hostname | | 存放本资料的主机名字 |
| | | Contact | | 管理域的管理员的邮件地址，因为“@”在文件中有特殊含义，所以邮件地址 abc@xyz.com 写为 abc.xyz.com |
| | | 时间数据字段 | Serial | 本区信息文件的版本号（文件修改后要将其值加 1） |
| | | | Refresh | 辅助域名服务器多长时间更新数据库 |
| | | | Retry | 若辅助域名服务器更新数据失败，多长时间再试 |
| | | | Expire | 若辅助域名服务器无法从主机上更新数据，原有的数据何时失效 |
| | | | Minimum | 若资源记录栏未设定 ttl，则以这里提供的时间为准 |

2）区文件指令

表 8-4 列出了可以在区文件中使用的 4 个区文件指令。

表 8-4　区文件指令

| 用　途 | 区文件指令 | 说　明 |
|---|---|---|
| 简化区文件结构 | $INCLUDE | 读取一个外部文件并包含它 |
| | $GENERATE | 用来创建一组 NS、CNAME 或 PTR 类型的 RR |
| 由资源记录使用的值 | $ORIGIN | 设置管辖源 |
| | $TTL | 为没有定义精确的生存期的 RR 定义默认的 ttl 值 |

### 3．DNS 服务器配置实例

为了方便读者对DNS服务器配置文件的使用有个详细的了解，下面将针对一个实际的配置文件例子来进行讲解。我们虚构了一个域tsinghua.com来举例说明主服务器的配置，下面是定义tsinghua.com域的主服务器的named.conf文件：

```
// generated by named-bootconf.pl
```

```
options {
 directory "/var/named";
 /*
 * If there is a firewall between you and nameservers you want
 * to talk to, you might need to uncomment the query-source
 * directive below. Previous versions of BIND always asked
 * questions using port 53, but BIND 8.1 uses an unprivileged
 * port by default.
 */
 // query-source address * port 53;
 };

// a caching only nameserver config
 //
 zone "." {
 type hint;
 file "named.ca";
 };
 zone "tsinghua.com"{
 type master;
 file "tsinghua.com";
 };

zone "0.0.127.in-addr.arpa" {
 type master;
 file "named.local";
 };

zone "132.211.in-addr.arpa"{
 type master;
 file "named.rev";
 };
```

上面的例子中第一个master告诉我们这是tsinghua.com域的主服务器。该域的数据是从named.hosts文件中加载的。在这个例子中，我们将文件名named.hosts作为区文件名。第三个master语句指向能将IP地址 211.132.0.0 映射为主机名的文件。它假定本地服务器是反向域132.211.in-addr.arpa的主服务器，该域的数据从文件named.rev中加载。

除了定义上述的主文件外，还需要定义以下的区文件（/var/named/tsinghua.com）：

```
 $TTL                                86400
 $ORIGIN tsinghua.com.
```

```
@                                          1D IN SOA   @ root (
                                           42      ; serial (d. adams)
                                           3H      ; refresh
                                           15M     ; retry
                                           1W      ; expiry
                                           1D )    ; minimum

                                           @ IN NS @
                                           @ IN A      127.0.0.1
                                           www IN A    211.132.211.80
                                           ftp IN A    211.132.211.68
                                           web IN CNAME    www
```

### 8.2.2 使用 Dlint 工具进行 DNS 配置文件检查

Dlint 是一个专门检查 DNS 配置文件开放源代码的软件，用户可以从网站 http://www.domtools.com上自行下载安装，目前该网站上的最新版本为Dlint1.4.1。需要注意：使用该软件前要求系统安装支持Perl语言和Dig命令（BIND中的一个软件包）的相关软件包。

Dlint软件的安装步骤如下，系统会将Dlint安装在/usr/bin/目录下。

（1）解压缩软件包：

```
#tar dlint1.4.1.tar.gz
```

（2）切换到解压缩的目录下，执行安装命令：

```
#cd dlint1.4.1
#make
```

Dlint主要针对DNS配置文件进行如下检查。

- 检查配置文件是否存在拼写错误。
- 检查配置文件中是否有 A（Address）记录的主机名称都有配套的 PTR（反向解析记录的简称）记录。如果有 A 记录的主机名称没有 PTR，则配置文件不能通过。另外，Dlint 可以在用户配置文件中为 A 记录查找丢失的 PTR 记录。
- 记录 in-addr.arpa 区的每一条 PTR 记录是否有对应的 A 记录存在，并以递归的方式检查子区，查找它们的配置问题。

使用Dlint工具进行DNS配置文件检查的运行结果如下：

```
#dlint localhost.localhost
;; dlint version 1.4.1, Copyright (C) 1998 Paul A. Balyoz <pab@domtools.com>
;;     Dlint comes with ABSOLUTELY NO WARRANTY.
;;     This is free software, and you are welcome to redistribute it
;;     under certain conditions.  Type 'man dlint' for details.
;; command line: /usr/local/bin/dlint localhost.localhost
;; flags: normal-domain recursive.
;; using dig version 9.2.1
```

```
;; run starting: 涓? 4 鏈?15 07:08:18 CST 2009
;; ============================================================
;; Now linting localhost.localhost
ERROR: no name servers found for domain localhost.localhost
That domain is probably not a zone.  Remove the leftmost portion of the name and
try again.
;; ============================================================
;; dlint of localhost.localhost run ending with errors.
;; run ending: 涓? 4 鏈?15 07:08:18 CST 2009
```

### 8.2.3 使用命令检验 DNS 功能

1．nslookup 命令

nslookup命令是用来验证DNS功能以及故障的一种非常简便有效的工具，通过该命令可以对用户搭建的DNS服务器或者是公用的服务器的功能进行有效验证。该命令不但可以在Linux下使用，而且也可以在Windows系列操作系统中使用。如果该命令能够成功返回信息，也就是能够通过域名从DNS服务器得到需要解析的IP地址信息，或者通过提供IP地址信息得到域名信息，那么，就表明该DNS是正常运行的。

下面的例子给出使用nslookup命令向DNS查询www.sohu.com域名IP地址的应用场景，验证表明DNS功能正常：

```
//正向查询
#nslookup www.google.com
Note:  nslookup is deprecated and may be removed from future releases.
Consider using the 'dig' or `host' programs instead.  Run nslookup with
the '-sil[ent]' option to prevent this message from appearing.
Server:        10.2.13.18
Address:       10.2.13.18#53

Non-authoritative answer:
www.google.com  canonical name = www.l.google.com.
www.l.google.com canonical name = www-china.l.google.com.
Name:   www-china.l.google.com
Address: 72.14.235.99
Name:   www-china.l.google.com
Address: 72.14.235.104
Name:   www-china.l.google.com
Address: 72.14.235.147
//反向查询
#nslookup 72.14.235.99
```

```
Note:  nslookup is deprecated and may be removed from future releases.
Consider using the 'dig' or 'host' programs instead.  Run nslookup with
the '-sil[ent]' option to prevent this message from appearing.
Server:        10.2.13.18
Address:       10.2.13.18#53

Non-authoritative answer:
99.235.14.72.in-addr.arpa name = tw-in-f99.google.com.

Authoritative answers can be found from:
235.14.72.in-addr.arpa  nameserver = ns3.google.com.
235.14.72.in-addr.arpa  nameserver = ns1.google.com.
235.14.72.in-addr.arpa  nameserver = ns4.google.com.
235.14.72.in-addr.arpa  nameserver = ns2.google.com.
ns1.google.com  internet address = 216.239.32.10
ns2.google.com  internet address = 216.239.34.10
ns3.google.com  internet address = 216.239.36.10
ns4.google.com  internet address = 216.239.38.10
```

2. dig命令

dig（域信息搜索器）命令是一个用于询问DNS域名服务器的灵活的工具。它执行DNS搜索，显示从受请求的域名服务器返回的答复。多数DNS管理员利用dig作为DNS问题的故障诊断，因为它灵活性好、易用、输出清晰。虽然通常情况下dig使用命令行参数，但它也可以按批处理模式从文件读取搜索请求。不同于早期版本，dig的BIND9 实现允许从命令行发出多个查询。除非被告知请求特定域名服务器，dig将尝试/etc/resolv.conf中列举的所有服务器。当未指定任何命令行参数或选项时，dig将对“.”（根）执行DNS查询。

值得一提的是：dig命令比nslookup命令获取的信息更加详细和全面，因此在Linux下建议用户采用该命令替代nslookup命令进行查询和验证DNS的功能（nslookup命令运行中也提供了“Note: nslookup is deprecated and may be removed from future releases.Consider using the 'dig' or 'host' programs instead.”字样进行提示）。

另外，鉴于dig命令的强大功能，它提供了几十个参数选项供用户使用，具体情况可以使用man命令进行查阅，这不是本书的重点内容。下面仅仅给出两个例子来进行简单演示和说明。

（1）运行dig命令获得根DNS信息：

```
#dig


; <<>> DiG 9.2.1 <<>>
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41868
```

```
;; flags: qr rd ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 14

;; QUESTION SECTION:
;.                          IN     NS

;; ANSWER SECTION:
.                    290515  IN     NS     K.ROOT-SERVERS.NET.
.                    290515  IN     NS     J.ROOT-SERVERS.NET.
.                    290515  IN     NS     D.ROOT-SERVERS.NET.
.                    290515  IN     NS     M.ROOT-SERVERS.NET.
.                    290515  IN     NS     C.ROOT-SERVERS.NET.
.                    290515  IN     NS     G.ROOT-SERVERS.NET.
.                    290515  IN     NS     A.ROOT-SERVERS.NET.
.                    290515  IN     NS     E.ROOT-SERVERS.NET.
.                    290515  IN     NS     H.ROOT-SERVERS.NET.
.                    290515  IN     NS     F.ROOT-SERVERS.NET.
.                    290515  IN     NS     L.ROOT-SERVERS.NET.
.                    290515  IN     NS     B.ROOT-SERVERS.NET.
.                    290515  IN     NS     I.ROOT-SERVERS.NET.

;; ADDITIONAL SECTION:
D.ROOT-SERVERS.NET.    109756  IN     A      128.8.10.90
E.ROOT-SERVERS.NET.    109756  IN     A      192.203.230.10
F.ROOT-SERVERS.NET.    282557  IN     A      192.5.5.241
F.ROOT-SERVERS.NET.    282557  IN     AAAA    2001:500:2f::f
G.ROOT-SERVERS.NET.    109756  IN     A      192.112.36.4
H.ROOT-SERVERS.NET.    539017  IN     A      128.63.2.53
H.ROOT-SERVERS.NET.    541758  IN     AAAA    2001:500:1::803f:235
I.ROOT-SERVERS.NET.    109756  IN     A      192.36.148.17
J.ROOT-SERVERS.NET.    452944  IN     A      192.58.128.30
J.ROOT-SERVERS.NET.    455358  IN     AAAA    2001:503:c27::2:30
K.ROOT-SERVERS.NET.    103767  IN     A      193.0.14.129
L.ROOT-SERVERS.NET.    103467  IN     A      199.7.83.42
L.ROOT-SERVERS.NET.    103467  IN     AAAA    2001:500:3::42
M.ROOT-SERVERS.NET.    109072  IN     A      202.12.27.33

;; Query time: 50 msec
;; SERVER: 10.2.13.18#53(10.2.13.18)
;; WHEN: Wed Apr 15 06:39:20 2009
```

```
;; MSG SIZE  rcvd: 500
```

（2）正向解析域名www.google.com：

```
#dig www.google.com

; <<>> DiG 9.2.1 <<>> www.google.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53325
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 7, ADDITIONAL: 7

;; QUESTION SECTION:
;www.google.com.              IN      A

;; ANSWER SECTION:
www.google.com.        600893  IN      CNAME   www.l.google.com.
www.l.google.com.      292     IN      CNAME   www-china.l.google.com.
www-china.l.google.com. 81      IN      A       72.14.235.99
www-china.l.google.com. 81      IN      A       72.14.235.104
www-china.l.google.com. 81      IN      A       72.14.235.147

;; AUTHORITY SECTION:
l.google.com.          78559   IN      NS      a.l.google.com.
l.google.com.          78559   IN      NS      b.l.google.com.
l.google.com.          78559   IN      NS      d.l.google.com.
l.google.com.          78559   IN      NS      f.l.google.com.
l.google.com.          78559   IN      NS      c.l.google.com.
l.google.com.          78559   IN      NS      e.l.google.com.
l.google.com.          78559   IN      NS      g.l.google.com.

;; ADDITIONAL SECTION:
g.l.google.com.        66906   IN      A       74.125.95.9
a.l.google.com.        559     IN      A       209.85.139.9
b.l.google.com.        79061   IN      A       74.125.45.9
c.l.google.com.        64857   IN      A       64.233.161.9
d.l.google.com.        72926   IN      A       74.125.77.9
e.l.google.com.        65021   IN      A       209.85.137.9
f.l.google.com.        64805   IN      A       72.14.235.9
```

```
;; Query time: 21 msec
;; SERVER: 10.2.13.18#53(10.2.13.18)
;; WHEN: Wed Apr 15 06:28:57 2009
;; MSG SIZE  rcvd: 348
```

### 8.2.4 配置辅助域名服务器进行冗余备份

辅助服务器可从主服务器中复制一整套域信息。区文件是从主服务器中复制出来的，并作为本地磁盘文件存储在辅助服务器中，这种复制称为“区文件复制”。在辅助域名服务器中有一个所有域信息的完整拷贝，可以有权威地回答对该域的查询。因此，辅助域名服务器也称作权威性服务器。配置辅助域名服务器不需要生成本地区文件，因为可以从主服务器中下载该区文件。

辅助服务器的配置与主服务器的配置不同，它使用slave语句代替master语句。slave语句指向用作域信息源的远程服务器，以替代本地磁盘文件。下面的named.conf文件可以配置tsinghua.com域的辅助服务器：

```
 // generated by named-bootconf.pl

options {
 directory "/var/named";
 /*
 * If there is a firewall between you and nameservers you want
 * to talk to, you might need to uncomment the query-source
 * directive below. Previous versions of BIND always asked
 * questions using port 53, but BIND 8.1 uses an unprivileged
 * port by default.
 */
 // query-source address * port 53;
 };

//
 // a caching only nameserver config
 //
 zone "." {
 type hint;
 file "named.ca";
 };

zone "0.0.127.in-addr.arpa" {
 type master;
 file "named.local";
```

```
};
zone "tsinghua.com"{
type slave;
file "named.hosts";
masters {211.132.10.3;};
};
zone "132.211.in-addr.arpa"{
type slave;
file "named.rev";
masters {211.132.10.3;};
};
cache . named.ca
secondary vbrew.com 211.132.10.3 named.hosts
secondary 132.211.in-addr.arpa 211.132.10.3 named.rev
primary 0.0.127.in-addr.arpa named.local
```

第一个slave语句是使这个服务器成为vbrew.com的辅助服务器。它告诉named从IP地址为211.132.10.3 的服务器中下载tsinghua.com的信息，并将其数据保存在/var/named/named.hosts文件中。如果该文件不存在，named就创造一个，并从远程服务器中取得区数据，然后将这些数据写入新创建的文件中。如果存在该文件，named就要检查远程服务器，以了解该远程服务器的数据是否不同于该文件中的数据，如果数据有变化，它就下载更新后的数据，用新数据覆盖该文件的内容；如果数据没有变化，named就加载磁盘文件的内容，不必做麻烦的区转移工作。将一个数据库拷贝到本地磁盘文件中，就不必每次引导主机时都要转移区文件；只有当数据修改时，才进行这种区文件的转移工作。该配置文件中的下一行表示该本地服务器也是反向域132.211.in-addr.arpa的一个辅助服务器，而且该域的数据也从 211.132.10.3 中下载。该反向域的数据存储在named.rev中。

### 8.2.5 配置高速缓存服务器缓解 DNS 访问压力

高速缓存服务器可运行域名服务器软件，但是没有域名数据库软件。它从某个远程服务器取得域名服务器每次查询的结果，一旦取得一个，就将它放在高速缓存中，以后查询相同的信息时就用它予以回答。高速缓存服务器不是权威性服务器，因为它提供的所有信息都是间接信息。对于高速缓存服务器只需要配置一个高速缓存文件，但最常见的配置还包括一个回送文件，这或许是最常见的域名服务器配置。

配置高速缓存域名服务器很简单。必须有named.conf和named.ca文件，通常也要用到named.local文件。下面是用于高速缓存服务器的named.conf文件的例子：

```
// generated by named-bootconf.pl
options {
directory "/var/named";
/*
* If there is a firewall between you and nameservers you want
```

```
 * to talk to, you might need to uncomment the query-source
 * directive below. Previous versions of BIND always asked
 * questions using port 53, but BIND 8.1 uses an unprivileged
 * port by default.
 */
 // query-source address * port 53;
 };


 // a caching only nameserver config
 //


 // a caching only nameserver config
 //
 zone "." {
 type hint;
 file "named.ca";
 };

zone "0.0.127.in-addr.arpa" {
 type master;
 file "named.local";
```

directory这一行告诉named到哪里去找寻文件。所有其后命名的文件都将是相对于此目录的。该文件告诉named去维持一个域名服务器响应的高速缓存，并利用named.ca文件的内容去初始化该高速缓存。该高速缓存初始化文件的名字可以是任何名字，但一般使用/var/named/named.ca。并不是在该文件中使用一个hint语句就能使它成为高速缓存配置，几乎每一种服务器的配置都要用到cache语句，而是因为没有master和slave语句才使它成为一个高速缓存配置。

但是，在我们这个例子中却有一个master语句。事实上，几乎在每一个高速缓存的配置文件中都有这一个语句，它将本地服务器定义为自己的回送域的主服务器，并假定该域的信息存储在named.local文件中。这个回送域是一个in-addr.arpa域（in-addr.arpa域用于指定逆向解析，或IP地址到DNS名字解析），它将地址 127.0.0.1 映射为名字localhost。转换自己的回送地址对于大多数人都是有意义的，因为大多数的named.conf文件都包含这一项。

在大多数高速缓存服务器的配置文件中，这种directory、master和hint语句是唯一使用的语句，但也可以增加其他的语句，forwarders和slave等语句都可以使用。

### 8.2.6　配置 DNS 负载均衡

DNS负载均衡技术是在DNS服务器中为同一个主机名配置多个IP地址，在应答DNS查询时，DNS服务器对每个查询将以DNS文件中主机记录的IP地址按顺序返回不同的解析结果，将客户端

的访问引导到不同的机器上去，使得不同的客户端访问不同的服务器，从而达到负载均衡的目的。

我们通过下面的例子来进行介绍。现假设有三台服务器来应对www.tsinghua.com的请求。在采用Linux系统上实现起来比较简单，只需在该区域文件的数据记录中添加类似下面的资源记录即可：

```
www1                    IN      A       210.113.1.1
www2                    IN      A       210.113.1.2
www3                    IN      A       210.113.1.3
www                     IN      CNAME   www1
www                     IN      CNAME   www2
www                     IN      CNAME   www3
```

上述六条资源记录的具体含义为：在DNS服务器中为www.tsinghua.com设定了三台服务器响应客户的访问请求。这三台服务器分别为www1、www2 和www3，而它们均为www服务器的别名。因此，在访问www服务器时，DNS服务器将依次循环地将访问请求均衡到三台服务器中去，以达到负载均衡的目的。

### 8.2.7　限制名字服务器递归查询功能

关闭递归查询可以使名字服务器进入被动模式，它在向外部的DNS发送查询请求时，只会回答自己授权域的查询请求，而不会缓存任何外部的数据，所以不可能遭受缓存中毒攻击，但是这样做也有负面的效果，降低了DNS的域名解析速度和效率。

以下语句仅允许 172.168.10 网段的主机进行递归查询：

```
allow-recusion {172.168.10.3/24; }
```

### 8.2.8　限制区传送（zone transfer）

如果没有限制区传送，那么DNS服务器允许对任何人都进行区域传输，因此网络架构中的主机名、主机IP列表、路由器名和路由IP列表，甚至包括各主机所在的位置和硬件配置等情况都很容易被入侵者得到在DNS配置文件中通过设置来限制允许区传送的主机，从一定程度上能减轻信息泄露。但是，需要提醒用户注意的是：即使封锁整个区传送也不能从根本上解决问题，因为攻击者可以利用DNS工具自动查询域名空间中的每一个IP地址，从而得知哪些IP地址还没有分配出去，利用这些闲置的IP地址，攻击者可以通过IP欺骗伪装成系统信任网络中的一台主机来请求区传送。

以下语句仅允许IP地址为172.168.10.1和172.168.10.2的主机能够同DNS服务器进行区域传输：

```
acl list {
  221.3.131.5;
  221.3.131.6;
zone "test.com" {
   type master;
   file "test.com ";
   allow-transfer { list; };
};
};
```

### 8.2.9　限制查询（query）

如果任何人都可以对DNS服务器发出请求，那么这是不能接受的。限制DNS服务器的服务范围很重要，可以把许多入侵者拒之门外。修改BIND的配置文件：/etc/named.conf，加入以下内容即可限制只有 210.10.0.0/8 和 211.10.0.0/8 网段的查询本地服务器的所有区信息，可以在options语句里使用如下的allow-query子句：

```
options {
allow-query { 210.10.0.0/8; 211.10.0.0/8;};
};
```

### 8.2.10　分离 DNS（split DNS）

采用split DNS（分离DNS）技术把DNS系统划分为内部和外部两部分，外部DNS系统位于公共服务区，负责正常对外解析工作；内部DNS系统则专门负责解析内部网络的主机，当内部要查询Internet上的域名时，就把查询任务转发到外部DNS服务器上，然后由外部DNS服务器完成查询任务。把DNS系统分成内、外两个部分的好处在于Internet上其他用户只能看到外部DNS系统中的服务器，而看不见内部的服务器，而且只有内、外DNS服务器之间才交换DNS查询信息，从而保证了系统的安全性。并且，采用这种技术可以有效地防止信息泄露。

在BIND 9 中可以使用view语句进行配置分离DNS。view语句的语法为：

```
view view_name {
    match-clients { address_match_list };
    [ view_option; ...]
    zone_statement; ...
};
```

其中：

- match-clients：该子句非常重要，它用于指定谁能看到本 view。可以在 view 语句中使用一些选项。
- zone_statement：该子句指定在当前 view 中可见的区声明。如果在配置文件中使用了 view 语句，则所有的 zone 语句都必须在 view 中出现。对同一个 zone 而言，配置内网的 view 应该置于外网的 view 之前。

下面是一个使用view语句的例子，它来自于BIND9 的标准说明文档：

```
view  "internal" {
  match-clients { our-nets; };            // 匹配内网客户的访问
  recursion yes;                          // 对内网客户允许执行递归查询
  zone "example.com" {                    // 定义内网客户可见的区声明
    type master;
    file "example.com.hosts.internal";
  };
};
view  "external" {
```

```
   match-clients { any; };                  // 匹配 Internet 客户的访问
   recursion no;                            // 对 Internet 客户不允许执行递归查询
   zone "example.com" {                     // 定义 Internet 客户可见的区声明
      type master;
      file "example.com.hosts.external";
   };
};
```

接下来，需要在example.com.hosts.internal中创建内网客户可见的区文件，并在example.com.hosts.external中创建Internet客户可见的区文件。该区文件的编写可以根据用户的实际情况。

### 8.2.11 隐藏 BIND 的版本信息

通常软件的漏洞和风险信息是和特定版本相关的，因此版本号是黑客进行攻击所需要搜集的最有价值的信息之一。黑客使用dig命令可以查询BIND的版本号，然后黑客就能够通过版本号查询这个软件有哪些漏洞，并寻求相应的工具来针对该漏洞进行攻击。因此，随意公开BIND版本号是不明智的，具有很大的风险。其实，隐藏BIND版本号比较简单，只需要修改配置文件/etc/named.conf，在options部分添加version声明将BIND的版本号信息覆盖即可。使用下面的配置声明将BIND版本号覆盖，当有人请求版本信息时，将无法得到有用的版本信息：

```
options {
version "Unkown"
};
```

### 8.2.12 使用非 root 权限运行 BIND

在Linux内核 2.3.99 以后的版本中，可以以-u选项以非root权限运行BIND。该命令表示以nobody用户身份运行BIND，使用nobody身份运行能够降低缓冲区溢出攻击所带来的危险。命令如下：

```
#/usr/local/sbin/named -u nobody
```

### 8.2.13 删除 DNS 上不必要的其他服务

网络服务是造成系统安全的重要原因，常见的DoS攻击、弱脚本攻击以及缓冲区溢出攻击都是由于系统存在网络服务所引起的。在安装DNS运行所依赖的操作系统前，就应该确定在系统中运行的服务的最小集合，创建一个DNS服务器系统就不应该安装Web、POP、gopher、NNTP News等服务。建议不安装以下软件包：①X-Windows及相关的软件包；②多媒体应用软件包；③任何不需要的编译程序和脚本解释语言；④任何不用的文本编辑器；⑤不需要的客户程序；⑥不需要的其他网络服务。

### 8.2.14 合理配置 DNS 的查询方式

DNS的查询方式有两种，递归查询和迭代查询。合理配置这两种查询方式，能够在实践中取得较好的效果。

其中，递归查询是最常见的查询方式，工作方式是：域名服务器将代替提出请求的客户机（下级DNS服务器）进行域名查询，若域名服务器不能直接回答，则域名服务器会在域各树中的各分支的上下进行递归查询，最终将返回查询结果给客户机，在域名服务器查询期间，客户机将完全处于等待状态。具体流程示意如图 8-2 所示。

迭代查询又称重指引查询、其工作方式为：当服务器使用迭代查询时能够使其他服务器返回一个最佳的查询点提示或主机地址，若此最佳的查询点中包含需要查询的主机地址，则返回主机地址信息，若此时服务器不能够直接查询到主机地址，则是按照提示的指引依次查询，直到服务器给出的提示中包含所需要查询的主机地址为止，一般的，每次指引都会更靠近根服务器（向上），查寻到根域名服务器后，则会再次根据提示向下查找。具体流程示意如图 8-3 所示。

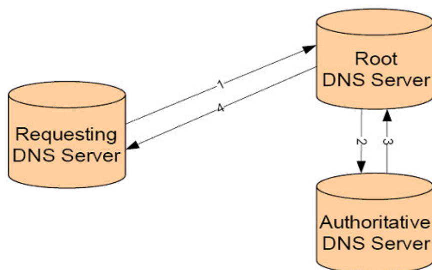



图 8-2　DNS 递归查询模式示意

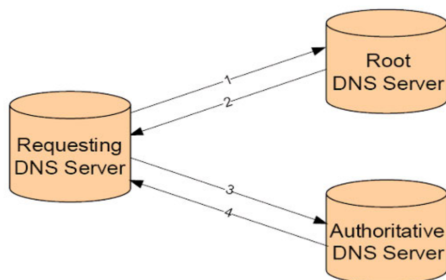



图 8-3　DNS 迭代查询模式示意

综合上面两点，我们可以看出来，递归查询就是客户机会等待最后结果的查询，而迭代查询是客户机等到的不一定是最终的结果，而可能是一个查询提示。因而存在如下两个问题。

- 二级 DNS 向一级 DNS 发起递归查询，会对一级 DNS 造成性能压力，所有跨域查询都要经过一级 DNS 响应给对应的二级 DNS。
- 二级 DNS 向一级 DNS 发起递归查询，再由一级向归属 DNS 发起递归的模式查询响应会有一定延时。

因此，有很多流量很大的DNS服务器禁止客户机使用递归查询，用这种方式来减轻服务器的流量。

## 8.2.15　使用 dnstop 监控 DNS 流量

在维护DNS服务器时，用户往往希望知道到底是哪些用户在使用DNS服务器，同时也希望能对DNS状态查询做一个统计，以及时地知道DNS的工作情况和状态。在传统的方式下，用户通常使用的是tcpdump等开源工具来进行抓包并通过查看 53 端口的流量来查看DNS数据包。由于tcpdump并没有针对DNS流量进行特殊定制，因此使用起来不是非常方便。因此，用户可以使用专用于DNS的dnstop工具查询DNS服务器的状态。

dnstop是一款非常优秀的开源软件，用户可以到网站http://dns.measurement-factory.com/tools/dnstop/src/上下载使用，目前该软件的最新版本为：dnstop-20090128.tar.gz。

由于该软件依赖tcpdump和pcap抓包库（libpcap）对网络上传输的数据包进行截获和过滤，所以用户需要确保系统安装相应软件后才能正常安装和使用dnstop。通常情况下，这两种必须的库都已经在系统中预装好了，使用下面的命令安装dnstop即可。

（1）解压缩源代码安装包：

```
#tar vxfz ddnstop-20090128.tar.gz
```

（2）切换到解压目录，并使用configure命令生成Makefile文件：

```
#cd dnstop-20040309
#./configure
```

（3）使用make命令进行安装：

```
#make
```

安装成功后，可以查看通过相应的网络接口来监控DNS网络流量，如下所示：

```
#./dnstop -s eth0
0 new queries, 6 total queries Tue Mar 26 19:35:23 2008
Sources count %
---------------- --------- ------
172.96.0.13 4 66.7
172.96.0.14 1 16.7
172.96.0.15 1 16.7
```

在运行dnstop的过程中，可以按S键、D键、T键、1 键、2 键，以及Ctrl+R组合键、Ctrl+X组合键等方式以交互方式来显示不同的信息。

- S：记录发送 DNS 查询的客户端 IP 地址列表。
- D：记录 DNS 查询的目的服务器的 IP 地址表。
- T：记录查询详细类型。
- 1：记录查询的顶级域名。
- 2：记录查询的二级域名。
- Ctrl+R：重新记录。
- Ctrl+X：退出。

更详细的信息可以使用dnstop-help命令进行查看。

## 8.3 企业电子邮件服务安全防护

### 8.3.1 安全使用 Sendmail Server

从使用的广泛程度和代码的复杂程度来讲，Sendmail是一款很优秀的邮件服务软件。几乎所有Linux的默认配置中都内置了这个软件，只需要设置好操作系统，它就能立即运转起来。但Sendmail的安全性较差，它在大多数系统中都是以root身份运行的，一旦邮件服务发生安全问题，将会对整个系统造成严重影响。同时在Sendmail开放之初，Internet用户数量及邮件数量都较少，使得Sendmail的系统结构并不适合较大的负载，对于高负载的邮件系统，需要对Sendmail进行复杂的调整。因此，在本节，我们将详细介绍有关安全使用该邮件服务器的方方面面。

#### 1. 安装最新版本的 Sendmail 服务器

要安全使用Sendmail服务器，需要下载和安装该服务器的最新版本。用户可以从网站http://www.sendmail.org/上取得最新版本的源代码sendmail-8.14.3.tar.gz，然后按照如下步骤进行

操作。

（1）执行以下命令，解压缩源码包：

```
#tar -zxvf sendmail-8.14.3.tar.gz
```

（2）由于Sendmail自带了一个编译程序Build，所以不用make命令，执行以下命令即可：

```
#cd sendmail-8.14.3/src
#./Build install
```

### 2. 使用 sendmail.mc 文件

Sendmail的配置十分复杂，其配置文件是sendmail.cf，位于/etc/mail目录下。由于sendmail.cf的语法深奥难懂，很少有人会直接去修改该文件来对sendmail服务器进行配置。我们一般通过m4宏处理程序来生成所需的sendmail.cf文件。创建过程中还需要一个模板文件，系统默认在/etc/mail目录下有一个sendmail.mc模板文件。

用m4 宏编译工具创建sendmail.cf文件比较方便，而且不容易出错，更可以避免某些带有安全漏洞或者过时的宏所造成的破坏。一个sendmail.mc模板的大致内容如下：

```
divert(-1)dnl
…
include('/usr/share/sendmail-cf/m4/cf.m4')dnl
VERSIONID('setup for Red Hat Linux')dnl
OSTYPE('Linux')dnl
…
dnl #
dnl define('SMART_HOST','smtp.your.provider')
dnl #
define('confDEF_USER_ID',''8:12'')dnl
define('confTRUSTED_USER', 'smmsp')dnl
dnl define('confAUTO_REBUILD')dnl
…
```

下面介绍sendmail.mc模板的语法组成。

- dnl：用来注释各项，同时 dnl 命令还用来标识一个命令的结束。
- divert(-1)：位于 mc 模板文件的顶部，目的是让 m4 程序输出时更加简洁一些。
- OSTYPE ('OperationSystemType')：定义使用的操作系统类型，显然这里应该用 Linux 代替 OperationSystemType，注意要用一个反引号和一个正引号把对应的操作系统类型括起来。
- define：定义一些全局设置，对于 Linux 系统，设置了 OSTYPE 之后，可以定义下面的一些全局参数，如果不定义，就使用默认值。下面给出例子：define（'ALIAS_FILE', '/etc/aliases'）：定义别名文件（alia file）的保存路径，默认是/etc/aliases；Define（'STATUS_FILE', '/etc/mail/statistics'）：定义 Sendmail 的状态信息文件。

因此，用户可以根据简单、直观的sendmail.mc模板来生成sendmail.cf文件，而无须直接编辑sendmail.cf文件。可以直接通过修改sendmail.mc模板来达到定制sendmail.cf文件的目的。这里介绍创建sendmail.cf文件的步骤。

（1）备份原有的sendmail.cf文件：

```
#cp /etc/mial/sendmail.cf /etc/mail/sendmail.cf.BAK
```

（2）生成sendmail.cf文件，根据sendmail.mc模板文件产生sendmail.cf配置文件，并导出到/etc/mail/目录下：

```
#m4 /etc/mail/sendmail.mc > /etc/mail/sendmail.cf
```

（3）重启 Sendmail 服务。

### 3. 使用 Access 数据库

访问数据库不但定义了什么主机或者IP地址可以访问本地邮件服务器，而且也定义了它们是属于哪种类型的访问。在定义中，主机可能会列出OK、REJECT、RELAY或者简单的通过Sendmail的出错处理程序检测一个给定的邮件错误。主机默认列出OK，允许传送邮件到主机，只要邮件的最后目的地是本地主机。列出REJECT将拒绝所有的邮件连接。如果带有RELAY选项的主机将被允许通过这个邮件服务器发送邮件到任何地方。

现在一般Linux发行版本中自带的Sendmail服务器都默认禁止其他不明身份的主机利用本地服务器投递邮件，这样就大大减少了被非法的主机利用本地服务器来投递垃圾邮件的可能。同样地，对于合法的主机，则可以使用本地服务器来投递邮件，这就要对Access数据库进行配置。

/etc/mail/access.db是一个散列表数据库，是使用/etc/mail/access文件产生的，该文件为纯文本文件，其格式如下：

```
地址 操作
```

其中，地址字段的格式说明如表 8-5 所示，操作字段的说明如表 8-6 所示。

表 8-5 地址字段格式说明

| 地址字段 | 含 义 |
| --- | --- |
| domain | 用户域内所有主机 |
| ip address | 网段内的或者特定的主机 |
| use@domain | 特定的邮件地址 |
| uer@ | 用户名为 user 的邮件 |

表 8-6 操作字段说明

| 操作字段 | 含 义 |
| --- | --- |
| OK | 无条件接收或者发送 |
| RELAY | 允许 SMTP 中继邮件 |
| REJECT | 拒绝接收并给出错误信息 |
| DISCARD | 丢弃邮件，但是不提供错误信息 |

下面给出使用上述地址和操作字段修改/etc/mail/access文件，从而生成/etc/mail/access.db数据库的操作步骤。

（1）使用vi编辑/etc/mail/access文件，并添加以下内容：

```
//允许 samsung.com 的域内所有主机使用服务器转发邮件
samsung.com     RELAY
```

```
//允许 IP 地址为 10.0.6.134 的主机使用服务器转发邮件
10.0.6.134       RELAY

//允许 192.168.10 网段内的主机使用服务器转发邮件
RELAY
//拒绝 info.com 的域内所有主机使用服务器
info.com        REJECT
```

（2）存盘退出，并且使用makemap命令生成/etc/mail/access.db数据库即可：

```
#makemap hash access.db <access
```

### 4. 配置带 SMTP 认证的 Sendmail 服务器

上面介绍了使用Access数据库来管理用户，合法地使用SMTP服务器进行邮件的传递。然而，由于用户的不断增多，并且很多用户都是在一个网段内，如果仅仅依靠上述的Access数据库，则很难有效地管理SMTP服务器的使用，那样会使得Access数据库规模增大，管理混乱，从而造成效率降低，甚至是出错。这样，既不能保障一些合法用户正常地使用邮件服务器的open relay功能，也给一些不法的用户提供了可乘之机，他们可以利用这来发送一些垃圾邮件和一些非法邮件。所以，非常有必要使用身份认证程序库，配合Sendmail服务器一起使用，对使用SMTP服务的用户进行身份认证，从而保证该服务的合法使用。

Cyrus SASL是Cyrus Simple Authentication and Security Layer的简写，其最大的功能是为应用程序提供了认证函数库。应用程序可以通过函数库所提供的功能定义认证方式，并让SASL通过与邮件服务器主机的沟通从而提供认证的功能。许多Linux版本都使用Cyrus SASL身份认证程序库，对用户进行身份认证。使用前要下载sasl库，该函数库提供了安全认证所需的函数，下载地址为：http://cyrusimap.web.cmu.edu/downloads.html#sasl，目前网上最新版本为cyrus-sasl-2.1.22.tar.gz。

下面详细介绍使用该身份认证程序库配置带认证的sendmail的步骤。

（1）通过使用下载的源代码安装包，进行如下安装：

```
//解压缩源代码安装包
#tar -xzvf cyrus-sasl-2.1.22.tar

//切换到安装目录
#cd cyrus-sasl-2.1.22

//预编译生成 Makefile 文件
#./configure -prefix=/usr --enable-login --with-pwcheck --with-digest

//编译
#make
```

```
//安装模块
#make install
```

（2）查看Sendmail与认证相关的配置：

```
#cat /usr/lib/sasl/sendmail.conf
pwcheck_metod:pam
```

（3）编辑sendmail.mc，修改和认证相关的配置内容，删除每行的dnl即可：

```
#cd /etc/mail
#vi sendmail.mc
//取消如下行的注释
TRUST_AUTH_MECH('EXTERNAL DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
define('confAUTH_MECHANISMS', 'EXTERNAL GSSAPI DIGEST-MD5 CRAM-MD5 LOGIN
PLAIN')dnl
```

（4）使用m4 命令生成cf文件：

```
#m4 sendmail.mc > sendmail.cf
```

（5）重新启动Sendmail服务器。

（6）测试sasl：

```
#sendmail -d0.1 -bv root | grep SASL
NETUNIX NEWDB NIS PIPELINING SASL SCANF STARTTLS TCPWRAPPERS
```

通过上述六个步骤的配置，便成功地完成了对带SMTP认证的Sendmail邮件服务器的配置。

### 8.3.2 安全使用 Postfix 电子邮件服务器

Postfix是一个由IBM资助、Wietse Venema负责开发的自由软件工程产物，它的目的就是为用户提供除Sendmail之外的邮件服务器选择。Postfix在快速、易于管理和提供尽可能的安全性方面都进行了较好的考虑。Postfix是基于半驻留、互操作的进程的体系结构，每个进程完成特定的任务，没有任何特定的进程衍生关系，使整个系统进程得到很好的保护。同时Postfix也可以和Sendmail邮件服务器保持兼容性以满足用户的使用习惯。

与Sendmail相比，Postfix最被人称道的地方就在于其配置文件的可读性很高。Postfix的主配置文件是/etc/postfix/main.cf。虽然该配置文件的内容比较多，但其中大部分内容都是注释（“#”号开头的行），真正需要自行定义的参数并不多。然而，为了对其进行安全配置，仍需要针对某些选项进行细心的设置。

在main.cf文件中，参数都是以类似变量的设置方法来设置的，这些参数的使用主要包含两部分的内容。

- 定义和声明变量：例如 definename = good-better-best。等号左边是变量的名称，等号右边是变量的值。
- 引用变量：可以在变量的前面加上符号“$”来引用该变量，如：myname = $ definename（相当于 definename = good-better-best）。

需要注意的是：等号两边需要有空格字符。此外，如果变量有两个以上的设置值，就必须用逗号“，”或者空格符“ ”将它们分开。

在熟悉了上述变量的定义和引用方法后，下面详细介绍如何安全、高效地配置Postfix服务器

的相关选项。

#### 1．设置 Postfix 服务监听的网络接口

默认情况下，inet_interfaces参数的值被设置为localhost，这表明只能在本地邮件主机上寄信。如果邮件主机上有多个网络接口，而又不想使全部的网络接口都开放Postfix服务，就可以用主机名指定需要开放的网络接口。不过，通常是将所有的网络接口都开放，以便接收从任何网络接口来的邮件，即将inet_interfaces参数的值设置为“all”，如下所示：

```
inet_interfaces = all
```

#### 2．安全设置可接收邮件的主机名称或域名

mydestination参数非常重要，因为只有当发来的邮件的收件人地址与该参数值相匹配时，Postfix才会将该邮件接收下来。通过该选项的设置可以过滤掉许多没有经过认证和授权的邮件，从而节省服务器的存储空间，以及节省用户的邮件处理时间。

举一个简单的例子，用户可以将该参数值设置为如下：

```
accept_domain = test.net
mydestination = $accept_domain
```

这就表明只要来信的收件人地址是X@test.net（其中X表示某用户在test.net域中的邮件账户名），Postfix都会接收这些邮件。而除此之外的邮件，Postfix都不会接收。

#### 3．安全设置可转发邮件的网络（IP 设置）

有关安全设置可转发邮件的网络可以使用mynetworks参数来设置。可将该参数值设置为所信任的某台主机的IP地址，也可设置为所信任的某个IP子网或多个IP子网（采用“，”或者“ ”分隔）。

比如，用户可以将mynetworks参数值设置为 172.168.96.0/24，则表示这台邮件主机只转发子网 172.168.96.0/24 中的客户端所发来的邮件，而拒绝为其他子网转发邮件：

```
mynetworks = 172.168.96.0/24
```

除了mynetworks参数外，还有一个用于控制网络邮件转发的参数是mynetworks-style，它主要用来设置可转发邮件网络的方式。通常有以下三种方式。

- class：在这种方式下，Postfix 会自动根据邮件主机的 IP 地址得知它所在的 IP 网络类型（即 A 类、B 类或是 C 类），从而开放它所在的 IP 网段。
- subnet：这是 Postfix 的默认值，Postfix 会根据邮件主机的网络接口上所设置的 IP 地址、子网掩码来得知所要开放的 IP 网段。
- host：在这种方式下，Postfix 只会开放本机。

通常，用户一般不需要设置mynetworks-style参数，而直接设置mynetworks参数。如果对这两个参数都进行了设置，那么mynetworks参数的设置有效。

#### 4．设置可转发邮件的网络（域名设置）

上面介绍的mynetworks参数是针对邮件来源的IP来设置的，而relay_domains参数则是针对邮件来源的域名或主机名来设置的。其实从原理上来说是一致的，不过是区分了IP地址和域名而已，不过，relay_domains还需要依赖DNS这个基础设施。

例如，用户可以将relay_domains参数值设置为test.net，则表示任何由域test.net发来的邮件都

会被认为是信任的，Postfix会自动对这些邮件进行转发，如下所示：

```
relay_domains = test.net
```

那么，要使它能在实际网络中更好地转发邮件，还必须进行相应的DNS设置。因此需要在该网络的DNS服务器上定义一个主区域test.net，并在该区域配置文件中定义以下记录：

```
//定义邮件服务器的 IP 地址
patterson.test.net. IN  A  172.168.96.254
//定义邮件服务器的别名
mail.test.net. IN CNAME  patterson.test.net.
//定义优先级别
test.net. IN MX  10 mail.test.net.
```

#### 5. Postfix 使用 SMTP 安全认证

如同前面所述的Sendmail服务器面临的邮件转发的问题，在Postfix服务器中同样也存在。为了避免这种情况的出现，Postfix默认不会对外开放转发功能，而仅对本机（localhost）开放转发功能。但是，在实际应用中，必须在Postfix主配置文件中通过设置mynetworks、relay_domains参数来开放一些所信任的网段或网域，否则该邮件服务器几乎没有什么用处。在开放了这些所信任的网段或网域后，还可以通过设置SMTP认证，对要求转发邮件的客户端进行用户身份（用户账户名与密码）验证。只有通过了验证，才能接收该用户寄来的邮件并帮助转发。同样，Postfix中目前比较常用的SMTP认证机制是通过Cyrus SASL包来实现的。

默认情况下，Postfix并没有启用SMTP认证机制。要想让Postfix启用SMTP认证，就必须对Postfix的主配置文件/etc/postfix/main.cf进行修改。用户需要在main.cf文件中添加以下有关SMTP认证的设置部分：

```
smtpd_sasl_auth_enable = yes
smtpd_sasl_local_domain = ''
smtpd_recipient_restrictions = permit_mynetworks
permit_sasl_authenticated, reject_unauth_destination
broken_sasl_auth_clients=yes
smtpd_client_restrictions = permit_sasl_authenticated
smtpd_sasl_security_options = noanonymous
```

其中，每个选项的具体含义如下。

- smtpd_sasl_auth_enable：指定是否要启用 SASL 作为 SMTP 认证方式。默认为不启用，这里必须将它启用，所以要将该参数值设置为 yes。
- smtpd_sasl_local_domain：如果采用 Cyrus-SASL 版进行认证，那么这里不做设置。
- smtpd_recipient_restrictions：表示通过收件人地址对客户端发来的邮件进行过滤。通常有以下几种限制规则。
    - permit_mynetworks：表示只要是收件人地址位于 mynetworks 参数中指定的网段就可以被转发邮件。
    - permit_sasl_authenticated：表示允许转发通过 SASL 认证的邮件。
    - reject_unauth_destination：表示拒绝转发含未信任的目标地址的邮件。

- broken_sasl_auth_clients：表示是否兼容非标准的 SMTP 认证。有一些 Microsoft 的 SMTP 客户端采用非标准的 SMTP 认证协议，只需将该参数设置为 yes 就可解决这类不兼容问题。
- smtpd_client_restrictions：表示限制可以向 Postfix 发起 SMTP 连接的客户端。如果要禁止未经过认证的客户端向 Postfix 发起 SMTP 连接，则可将该参数值设置为 permit_sasl_authenticated。
- smtpd_sasl_security_options：用来限制某些登录的方式。如果将该参数值设置为 noanonymous，则表示禁止采用匿名登录方式。

在完成上述设置后，必须使用命令/etc/init.d/postfix reload重新载入配置文件，或使用命令/etc/init.d/postfix restart重新启动Postfix服务，以使该配置生效。当然，这两个命令的具体使用需要根据不同的Linux版本来选用。

此外，由于当Postfix要使用SMTP认证时，会读取/usr/lib/sasl2/smtpd.conf文件中的内容，以确定所采用的认证方式，因此如果要使用saslauthd守护进程来进行密码认证时，就必须确保/usr/lib/sasl2/smtpd.conf文件中的内容为：

```
pwcheck_method: saslauthd
```

### 8.3.3 企业垃圾邮件防护

#### 1. 常用技术

目前，垃圾电子邮件已成为人们最头疼的问题之一。在Linux操作系统平台中，反击和过滤垃圾电子邮件是一件很重要的工作。下面介绍一些在Linux中广泛使用的防垃圾邮件技术。

- SMTP 用户认证技术。目前常见并十分有效的方法是，在邮件传送代理 MTA 上对来自本地网络以外的互联网的发信用户进行 SMTP 认证，仅允许通过认证的用户进行远程转发。这样既能够有效避免邮件传送代理服务器被垃圾邮件发送者所利用，又为出差在外或在家工作的员工提供了便利。如果不采取 SMTP 认证，那么在不牺牲安全的前提下，设立面向互联网的 Web 邮件网关也是可行的。此外，如果 SMTP 服务和 POP3 服务集成在同一服务器上，在用户试图发信之前对其进行 POP3 访问验证（POP before SMTP）就是一种更加安全的方法，但在应用的时候要考虑到当前支持这种认证方式的邮件客户端程序还不多。
- 逆向 DNS 解析。无论哪一种认证，其目的都是避免邮件传送代理服务器被垃圾邮件发送者所利用，但对于发送到本地的垃圾邮件仍然无可奈何。要解决这个问题，最简单有效的方法是对发送者的 IP 地址进行逆向名字解析。通过 DNS 查询来判断发送者的 IP 与其声称的名字是否一致，例如，其声称的名字为 mx.hotmail.com，而其连接地址为 20.200.200.200，与其 DNS 记录不符，则予以拒收。这种方法可以有效过滤掉来自动态 IP 的垃圾邮件，对于某些使用动态域名的发送者，也可以根据实际情况进行屏蔽。但是上面这种方法对于借助 Open Relay 的垃圾邮件依然无效。对此，更进一步的技术是假设合法的用户只使用本域具有合法互联网名称的邮件传送代理服务器发送电子邮件。例如，若发件人的邮件地址为 someone@yahoo.com，则其使用的邮件传送代理服务器的 Internet 名字应具有 yahoo.com 的后缀。这种限制并不符合 SMTP 协议，但在多数情况下是切实有效的。需要指出的是，逆向名字解析要进行大量的 DNS 查询。
- 实时黑名单过滤。以上介绍的防范措施对使用自身合法域名的垃圾邮件仍然无效。对

此比较有效的方法就是使用黑名单服务了。黑名单服务是基于用户投诉和采样积累而建立的、由域名或IP组成的数据库，最著名的是RBL、DCC和Razor等，这些数据库保存了频繁发送垃圾邮件的主机名字或IP地址，供MTA进行实时查询以决定是否拒收相应的邮件。但是，目前各种黑名单数据库难以保证其正确性和及时性。例如，北美的RBL和DCC包含了我国大量的主机名字和IP地址，其中有些是早期的Open Relay造成的，有些则是由于误报造成的。但这些问题迟迟得不到纠正，在一定程度上阻碍了我国与北美地区的邮件联系，也妨碍了我国的用户使用这些黑名单服务。

- 白名单过滤。白名单过滤是相对于上述的黑名单过滤来说的。其建立的数据库的内容和黑名单的一样，但其性质是：库中存在的都是合法的，不应该被阻断。同样，该过滤方法存在的缺点与黑名单类似，也是更新和维护难以达到实时，一些正常的、不为系统白名单所收集的邮件有可能被阻断。从应用的角度来说，在小范围内使用白名单是比较成功的。
- 内容过滤。即使使用了前面诸多环节中的技术，仍然会有相当一部分垃圾邮件漏网。对此情况，目前最有效的方法是基于邮件标题或正文的内容过滤。其中比较简单的方法是，结合内容扫描引擎，根据垃圾邮件的常用标题语、垃圾邮件受益者的姓名、电话号码、Web 地址等信息进行过滤。更加复杂但同时更具智能性的方法是，基于贝叶斯（Bayes）概率理论的统计方法所进行的内容过滤，该算法最早由Paul Graham提出（http://www.paulgraham.com/spam.html），并使用他自己设计的Arc语言实现。这种方法的理论基础是通过对大量垃圾邮件中常见关键词进行分析后得出其分布的统计模型，并由此推算目标邮件是垃圾邮件的可能性。这种方法具有一定的自适应、自学习能力，目前已经得到了广泛的应用。最有名的垃圾邮件内容过滤是Spamassassin，其使用Perl语言实现，集成了以上两种过滤方法，可以与当前各种主流的MTA集成使用。内容过滤是以上所有各种方法中耗费计算资源最多的，在邮件流量较大的场合，需要配合高性能服务器使用。另外，当前也有很多学者将人工神经网络、支持向量机、Winnow算法等及其学习的方法引入到内容过滤垃圾邮件的研究中来，并且取得了很好的效果。

### 2. 设置Sendmail防范垃圾邮件

1）限制邮件转发区域

所谓Relay就是指别人能使用提供该功能的SMTP邮件服务器给任何人发信，这样别有用心的垃圾发送者就可以使用这台邮件服务器大量发送垃圾邮件，而最后别人投诉的不是垃圾发送者，而是这台服务器，因此必须限制Relay这项功能。从8.9版本开始，默认的是不允许邮件转发（mail relay）的。最简单的允许邮件转发的方法是在文件/etc/mail/relay-domains中进行设置。该文件中列出的域名内的信件都允许通过本地服务器进行邮件转发。

为了更精确地设置，可以在sendmail.mc中添加以下几个参数允许被用来设置邮件转发。

- FEATURE（relay_hosts_only）：通常情况下，在文件/etc/mail/relay-domains 中列出的域名的主机都允许通过本地机转发，而该设置指定必须罗列出每个允许通过本机转发邮件的主机。
- FEATURE（relay_entire_domain）：该参数指示允许所有本地域通过本机进行邮件转发。
- FEATURE（access_db）：该参数指定利用哈希数据库/etc/mail/access 来决定是否允许某个主机通过本地进行邮件转发。

- FEATURE（blacklist_recipients）：若该参数被设置，则在决定是否允许某个主机转发邮件时同时查看邮件发送者地址和邮件接收者地址。
- FEATURE（rbl）：允许基于 maps.vix.com 由黑名单（Realtime Blackhole List）进行邮件拒绝，以防范垃圾邮件。
- FEATURE（accept_unqualified_senders）：允许接收发送者地址不包括域名的邮件，例如 user，而不是 user@B.NET。
- FEATURE（accept_unresolvable_domains）：通常来讲，Sendmail 拒绝接收发送者邮件地址指定的主机通过 DNS 不能解析的邮件，而该参数允许接收这种邮件。
- FEATURE（relay_based_on_MX）：该参数允许转发邮件接收者地址的 MX 记录指向本地的邮件，例如，本地接收到一个发送目的地址为 user@b.com 的邮件，而 b.com 域名的 MX 记录指向了本地机器，则本地机器就允许转发该邮件。

2）关闭Sendmail的Relay功能

如果不想进行细致的Relay限制，则可以采用关闭该功能的方法达到安全的目的。其方法就是，到Linux服务器的/etc/mail目录编辑access文件，去掉“*relay”之类的设置，只留“localhost relay”和“127.0.0.1 relay”两条即可。最后特别注意，修改access文件后还要使用命令使修改生效：

```
#makemap hash access.db < access。
```

3）在Sendmail中添加RBL功能

RBL（Realtime Blackhole List）是实时黑名单。国外有一些机构提供RBL服务，常用的RBL服务器地址有relays.ordb.org、dnsbl.njabl.org、bl.spamcop.net、sbl.spamhaus.org、dun.dnsrbl. net和dnsbl.sorbs.net。查询和删除RBL中的IP地址可以到http://openrbl.org/和http://ordb.org。RBL将收集到的专发垃圾邮件的IP地址加入他们的黑名单，只要在Sendmail中加入RBL认证功能，就会使邮件服务器在每次收信时都自动到RBL服务器上去查实，如果信件来源于黑名单，则Sendmail会拒收邮件，从而使用户少受垃圾邮件之苦。在Sendmail中添加RBL认证，需要对sendmail.mc添加以下内容：

```
FEATURE('dnsbl', 'relays.ordb.org', '"Email blocked using ORDB.org - see "')
```

最后执行“m4 sendmail.mc>sendmail.cf”和“service sendmail restart”两条命令，使有关Sendmail的修改生效。

4）打开Sendmail的SMTP认证功能

该功能的使用已经在前面一个小节专门做了介绍，所以在这里不再赘述。

### 3. 使用 SpamAssassin

SpamAssassin会评估传入的每个电子邮件，并给它分配一个数字，指出该电子邮件是垃圾邮件的可能性。数字越高，该电子邮件越有可能是垃圾邮件。基于其评级，可以过滤电子邮件。SpamAssassin在安装后就起作用，但可以修改其配置文件，使其更好地满足你的需要。

1）SpamAssassin的工作原理

SpamAssassin包括spamd守护进程和spamc客户端。虽然它包括SpamAssassin实用工具，但SpamAssassin文档建议使用spamc而非SpamAssassin来过滤邮件，因为spamc比SpamAssassin加载得更快。当SpamAssassin单独工作时，spamc调用spamd。spamd守护进程派生子进程；当spamd运行时，ps除了显示spamd父进程外还显示几个spamd子进程：

```
# ps -ef | grep spam
root 4254 1 0 14:17 ? 00:00:02 /usr/bin/spamd -d -c -m5 -H -r …
root 4256 4254 0 14:17 ? 00:00:00 spamd child
root 4257 4254 0 14:17 ? 00:00:00 spamd child
root 4689 4662 0 16:48 pts/1 00:00:00 grep --color=auto spam
```

spamc实用工具是一个过滤器：它从标准输入读取每个电子邮件，发送电子邮件到spamd进行处理，并把修改后的电子邮件写入到标准输出。spamd守护进程使用多种技术来识别垃圾邮件。

- Headeranalysis（标题分析）：检查疑似垃圾邮件的标题，有些垃圾邮件被人采用某种技巧处理后，可能会被误认为是合法的电子邮件。
- Text analysis（文本分析）：检查电子邮件正文中的垃圾邮件特征。
- Blacklists（黑名单）：检查名单，看看发件人是否在现有垃圾邮件发送者列表中。
- Database（数据库）：检查针对 Vipul's Razor（razor.sourceforge.net）的邮件签名，它是一个垃圾邮件跟踪数据库。

可以设置邮件服务器上的SpamAssassin，以使它处理传递到本地系统的所有电子邮件，然后再发送到用户。另外，每个用户可以从他们的邮件客户端运行SpamAssassin。无论哪种方式，本地系统必须运行spamd，并且必须通过该守护进程使用spamc过滤每封电子邮件。

2）安装SpamAssassin

安装如下软件包（默认都安装）。

- spamassassin
- procmail（需要在邮件服务器上运行 SpamAssassin）spamassassin init 脚本。

当系统进入多用户模式时，可运行chkconfig来启动spamassassin。

```
# chkconfig spamassassin on
```

启动 spamassassin:

```
# service spamassassin start
```

3）测试SpamAssassin

随着spamd的运行，向spamc发送一个字符串可以查看其工作原理：

```
$ echo "hi there" | spamc
X-Spam-Checker-Version: SpamAssassin 3.3.2-r929478 (2010-03-31) on sobell.com
X-Spam-Flag: YES
X-Spam-Level: ******
X-Spam-Status: Yes, score=6.9 required=5.0 tests=EMPTY_MESSAGE,MISSING_DATE,
MISSING_HEADERS,MISSING_MID,MISSING_SUBJECT,NO_HEADERS_MESSAGE,NO_RECEIVED,
NO_RELAYS autolearn=no version=3.3.2-r929478
X-Spam-Report:
* -0.0 NO_RELAYS Informational: message was not relayed via SMTP
* 1.2 MISSING_HEADERS Missing To: header
* 0.1 MISSING_MID Missing Message-Id: header
* 1.8 MISSING_SUBJECT Missing Subject: header
* 2.3 EMPTY_MESSAGE Message appears to have no textual parts and no
```

```
* Subject: text
* -0.0 NO_RECEIVED Informational: message has no Received headers
* 1.4 MISSING_DATE Missing Date: header
* 0.0 NO_HEADERS_MESSAGE Message appears to be missing most RFC-822
* headers
hi there
Subject: [SPAM]
X-Spam-Prev-Subject: (nonexistent)
```

它首先会显示Yes，即认定该邮件是垃圾邮件。SpamAssassin使用评级系统，给一封电子邮件分配一个匹配命中数。如果该电子邮件收到的命中数超过所需的数量（默认为 5.0），SpamAssassin则把它标记为垃圾邮件。字符串失败的原因是多方面的，都会在此状态行上列举。

以下列表是由SpamAssassin处理的一封真实垃圾邮件。它收到了 24.5 个命中，这几乎可以肯定是垃圾邮件。

```
X-Spam-Status: Yes, hits=24.5 required=5.0
tests=DATE_IN_FUTURE_06_12,INVALID_DATE_TZ_ABSURD,
MSGID_OE_SPAM_4ZERO,MSGID_OUTLOOK_TIME,
MSGID_SPAMSIGN_ZEROES,RCVD_IN_DSBL,RCVD_IN_NJABL,
RCVD_IN_UNCONFIRMED_DSBL,REMOVE_PAGE,VACATION_SCAM,
X_NJABL_OPEN_PROXY
version=2.55

X-Spam-Level: ************************
X-Spam-Checker-Version: SpamAssassin 2.55 (1.174.2.19-2003-05-19-exp)
X-Spam-Report: This mail is probably spam. The original message has been attached
along with this report, so you can recognize or block similar unwanted
mail in future. See http://spamassassin.org/tag/ for more details.
Content preview: Paradise SEX Island Awaits! Tropical 1 week vacations
where anything goes! We have lots of WOMEN, SEX, ALCOHOL, ETC! Every
man's dream awaits on this island of pleasure. [...]
Content analysis details: (24.50 points, 5 required)
MSGID_SPAMSIGN_ZEROES (4.3 points) Message-Id generated by spam tool (zeroes
variant)
INVALID_DATE_TZ_ABSURD (4.3 points) Invalid Date: header (timezone does not
exist)
MSGID_OE_SPAM_4ZERO (3.5 points) Message-Id generated by spam tool (4-zeroes
variant)
VACATION_SCAM (1.9 points) BODY: Vacation Offers
REMOVE_PAGE (0.3 points) URI: URL of page called "remove"
MSGID_OUTLOOK_TIME (4.4 points) Message-Id is fake (in Outlook Express format)
```

```
 DATE_IN_FUTURE_06_12 (1.3 points) Date: is 6 to 12 hours after Received: date
 RCVD_IN_NJABL (0.9 points) RBL: Received via a relay in dnsbl.njabl.org
 [RBL check: found 94.99.190.200.dnsbl.njabl.org.]
 RCVD_IN_UNCONFIRMED_DSBL  (0.5  points)  RBL:  Received  via  a  relay  in
unconfirmed.dsbl.org
 [RBL check: found 94.99.190.200.unconfirmed.dsbl.org.]
 X_NJABL_OPEN_PROXY (0.5 points) RBL: NJABL: sender is proxy/relay/formmail/
 spam-source
 RCVD_IN_DSBL (2.6 points) RBL: Received via a relay in list.dsbl.org
 [RBL check: found 211.157.63.200.list.dsbl.org.]
 X-Spam-Flag: YES
 Subject: [SPAM] re: statement
```

4）配置SpamAssassin

SpamAssassin在许多位置可查找配置文件，详细信息请参阅SpamAssassin手册页。最容易使用的配置文件是/etc/mail/spamassassin/local.cf，可以编辑这个文件来全局配置SpamAssassin。用户可以覆盖这些全局选项并在~/.spamassassin/user_prefs文件中添加自己的选项。可以把本节讨论的选项放在这些文件中的任何一个。

例如，可以配置SpamAssassin来重写评级为垃圾邮件的邮件主题行。配置文件中的rewrite_header关键字可控制这种行为。跟随这个关键字的Subject字告诉SpamAssassin重写主题行。从以下行删除#就可以启用这种行为：

```
 # rewrite_header Subject *****SPAM*****
```

required_score关键字指定：SpamAssassin认为它是垃圾邮件之前一封电子邮件必须获得的最低得分，默认值是 5.00。设置此关键字到一个更高的数值，就能使SpamAssassin把较少的电子邮件标记为垃圾邮件。

```
 required_score 5.00
```

有时标记为垃圾邮件地址的邮件并不是垃圾邮件，或者来自该地址的邮件并不应该标记为垃圾邮件。使用whitelist_from关键字可指定不应该被视为垃圾邮件的地址，blacklist_from用于指定应始终标记为垃圾邮件的地址：

```
 whitelist_from sams@example.com
blacklist_from *@spammer.net
```

可以在whitelist_from和blacklist_from行上指定多个地址，并用空格隔开。每个地址可以包含通配符。使用whitelist_from *@example.com将从example.com域发送电子邮件的每个人列入白名单。可以使用多个whitelist_from和blacklist_from行。

5）在邮件服务器上运行SpamAssassin

下面说明如何在邮件服务器上设置SpamAssassin，以使其处理传递到本地系统所有的电子邮件，然后才发送给用户。它显示了如何把procmail设置为MDA，以及如何让procmail通过spamc发送电子邮件。

首先确保MTA（sendmail）使用procmail 作为MDA。以下sendmail.mc中的前两行将指定procmail的路径、命令和标志。MAILER行定义procmail作为邮件程序。应该不必更改这些行。

```
define('PROCMAIL_MAILER_PATH', '/usr/bin/procmail')dnl
FEATURE(local_procmail, '', 'procmail -t -Y -a $h -d $u')dnl
MAILER(procmail)dnl
```

还要确保在服务器系统上安装procmail软件包。接下来，如果/etc/procmailrc配置文件不存在，则需要创建它，以使其所有者为root并拥有 644 权限和下列内容。如果它已经存在，可从以下文件给它追加最后两行：

```
$ cat /etc/procmailrc
DROPPRIVS=yes
:0 fw
| /usr/bin/spamc
```

这个文件的第一行确保尽可能少使用特权运行procmail。接下来的两行则实现了一个规则，即通过spamc用管道传输每个用户的来信。:0 告诉procmail后边是一条规则。f标志表示过滤器；w标志导致procmail等待该过滤器来完成并检查退出代码。最后一行指定/usr/bin/spamc实用工具作为过滤器。

有了这个文件，该服务器收到的所有本地邮件都通过SpamAssassin传递，SpamAssassin根据在全局配置文件中的选项为其评级。服务器系统上的用户账户可以覆盖在其~/.spamassassin/user_prefs文件中的全局的SpamAssassin配置设置。

当在服务器上运行SpamAssassin时，电子邮件的评级通常比较保守，以使得尽量少的电子邮件被标记为垃圾邮件。设置required_hits在 6～10 范围内通常比较合适。此外，不要自动删除任何电子邮件，因为这会阻止用户获得正常邮件。当服务器将邮件标记为可能的垃圾邮件时，用户可以手动或自动过滤该垃圾邮件，并决定如何处理。

#### 4. 客户端配置垃圾邮件防护功能

1）正确配置Foxmail收发邮件

在安装和启动了SMTP以及POP服务器之后，就可以使用其来收发邮件了，由于命令行方式的使用比较麻烦和需要相当的背景知识，所以现在普遍的方式是使用Windows下的一些邮件客户端软件来发送和接收邮件。

这些客户端软件主要有Foxmail、Outlook Express、Netscape、Eudora等，由于其他的客户端软件与Foxmail具有差不多的功能，下面主要介绍使用Foxmail这款典型的、最为常用的软件。该软件功能强大，使用灵活、方便。首先介绍如何来对其收发邮件的相应选项进行配置。

在安装好该软件的Foxmail 6.0 版本以后，就可以按照以下的步骤进行配置。

① 选择并单击菜单栏上的“邮箱（B）”选项，弹出下拉菜单，单击“新建邮箱账户（N）”选项，系统弹出“向导”对话框，如图 8-4 所示。向导将提示用户建立新的用户账户，在这里，建立一个新的用户为patterson。

② 单击“下一步”按钮，向导提示用户输入POP3 服务器地址、用户名、密码以及SMTP服务器地址，以方便发送和接收邮件，如图 8-5 所示。这里POP3 服务器地址为：pop3.test.net，用户名为：patterson，密码隐藏，SMTP服务器地址为：smtp.test.net。




图 8-4　Foxmail 新建用户向导

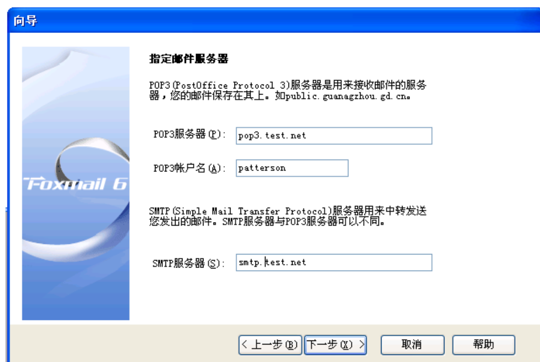



图 8-5　指定邮件服务器

③ 单击“下一步”按钮，如图 8-6 所示，并且单击“完成”按钮，完成账户的建立工作。配置完成后，用户就可以使用Foxmail方便地使用前面建立的test.net的SMTP以及POP3 服务器收发电子邮件了。

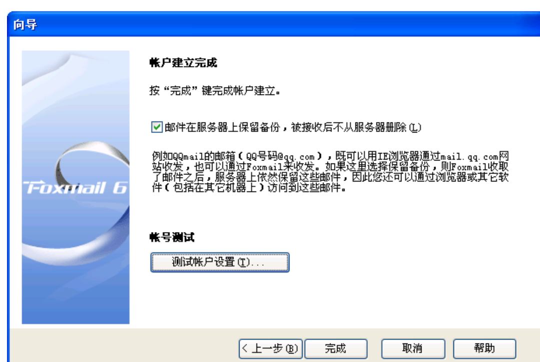



图 8-6　指定邮件服务器完成

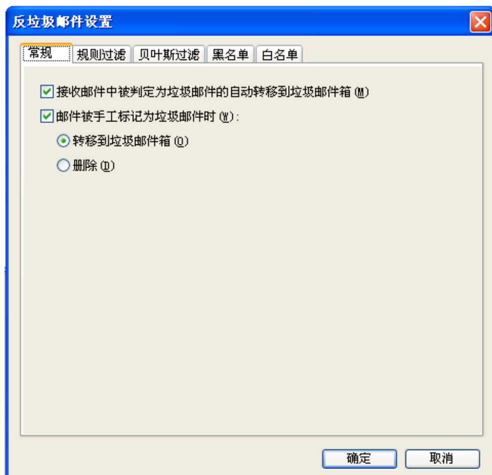
2）配置Foxmail垃圾邮件过滤高级功能

Foxmail提供了强大的垃圾邮件过滤功能，它基本上集成了前面介绍的所有垃圾邮件过滤技术。使用该客户端的用户可以结合其与sendmail邮件服务器高速使用，从而保证从一定程度上减轻垃圾邮件对自己的“骚扰”，具体的设置步骤如下。

① 选择并单击菜单栏上的“工具（T）”选项，弹出下拉菜单，单击“反垃圾邮件功能设置（Z）”选项，系统弹出“反垃圾邮件设置”对话框，如图 8-7 所示。

② 切换到“常规”选项卡，可以设置最简单的垃圾邮件转移处理功能。

③ 切换到“规则过滤”选项卡，可以设置垃圾邮件过滤的相关规则，如图 8-8 所示。



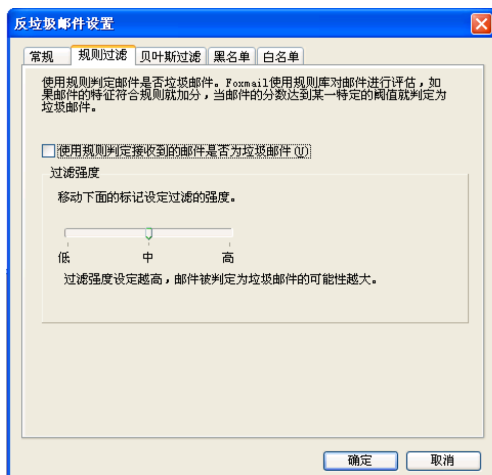




图 8-7　反垃圾邮件功能设置　　　　图 8-8　规则过滤

④ 切换到“贝叶斯过滤” 选项卡，可以设置基于贝叶斯人工智能方法的垃圾邮件过滤的相关方法，通过对现有用户标记的邮件的学习，Foxmail可以学习到垃圾邮件具有的关键字、IP地址、域名等信息，从而在邮件到来后，及时地、自动地为用户进行标记和判别，减少用户的处理时间，如图 8-9 所示。

⑤ 切换到“黑名单”选项卡，可以人工设置哪些邮件将直接列入黑名单而被Foxmail直接删除，从而在邮件到来后，及时地、自动地为用户进行标记和判别，同样减少用户的处理时间。在使用黑名单时特别要注意，千万不要把有用的邮件地址列入其中，否则有可能将给用户带来不可挽回的损失，如图 8-10 所示。

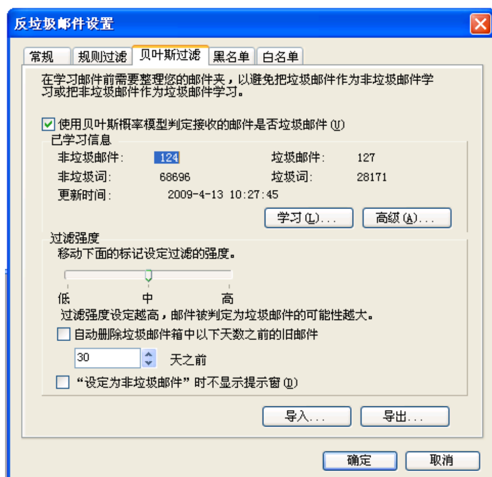



图 8-9　贝叶斯过滤

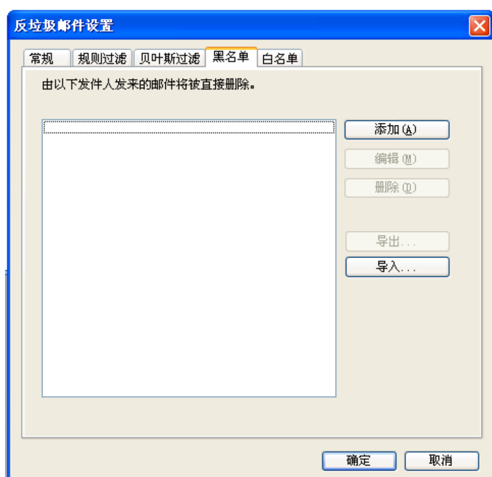



图 8-10　黑名单过滤

⑥ 切换到“白名单”选项卡，可以人工设置哪些邮件将直接列入白名单而不为Foxmail进行处理提示是否为垃圾邮件，从而交给用户作进一步处理，这样做同样可以减少用户的处理时间，如图 8-11 所示。

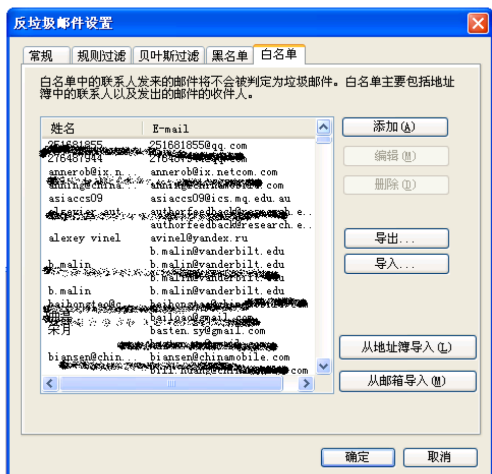



图 8-11　白名单过滤

# 第 9 章

# 未雨绸缪：企业级数据防护

## 本章导读

信息泄露的案例屡见不鲜，如汇丰银行离职员工造成的客户资料泄露、国内某大型造船厂发生的设计数据非法拷贝等事件，以及最近网络上的热门话题当数中国互联网“5000 万用户数据泄露”事件了，首先是知名技术网站 CSDN 被黑客泄露的 600 万明文账户密码信息，直接导致了许多用户的注册账号、注册密码及邮箱泄露，接着更是传人人网、多玩等多个中国大型网站也给“爆库”了。这一连串的事件使得众多网民人心惶惶、不断地更换密码。这也导致了大家对这些网站及其企业的信任危机。大家都在奔走相告，修改密码吧 ……

作为 500 强企业来说，数据加密及防护机制是 CIO、CSO 甚至是 CEO 必须认真考虑的一个问题。敏感数据泄露、公司机密被竞争对手所获得，这些后果都是致命的。作为业内的安全人士应冷静地看待这类事情，完全不必被这些事件的表面现象所惊吓。究其本质，是由于企业的数据加密和防护工作没有做到位，这不仅仅是技术的问题，更是管理的问题。其实，当前的数据加密和防泄露技术都比较成熟了，关键是没有落到实处。那么，500 强企业可以运用哪些数据加密和防泄露技术（包括工具）来从根本上避免这些问题的发生呢？本章将详细为读者进行介绍。

## 9.1 企业数据防护技术分析

在企业数据防护技术中，从本质上来说包括两大类，一类是数据加密；另一类则是数据防泄露。前者是为了解决数据的机密性和一致性问题，通俗来说就是防止不该看的人看到数据和篡改数据；而后者主要是为了解决数据的机密性问题，是为了定向地发送数据，控制数据的流通渠道，从本质上来说也是为了防止数据被非法或者不期望的用户所获得。

在很多用户账号泄露、机密数据泄密等安全事件中，很多事件就是因为以下几个原因造成的。

- 数据未加密：导致数据在传输通道（如有线网络、wifi 等）中或者物理服务器中被有意窃取和捕获，如 CSDN 的用户账号泄露事件等。如果数据加密了，黑客和不法用户即使捕获和窃取了加密的数据，要解开它们还是有相当难度的。
- 未做好数据防泄露工作：导致数据通过 USB、邮箱、即时消息等媒体和途径进行传播后，无法确保其仅能被有权限的人访问，从而导致间接泄密。

因此，根据这两个原因，本章将详细介绍企业如何使用开源工具来进行数据加密和数据防泄露工作。

## 9.2 数据加密技术原理

在加密技术中，明文是指未加密前的原始数据，密文是指加密后的数据。根据加密和解密时的密码不同，可以分为两种类型的加密、解密算法。

### 9.2.1 对称加密、解密

对称加密、解密是加密和解密使用相同的密码，有代表性的有DES、Blowfish、TEA、Base64。对称加密、解密的特点是运算相对非对称加密、解密简单，速度快，主要应用于需要加密大量数据的场合，例如游戏的资源文件加密，其示意如图 9-1 所示。

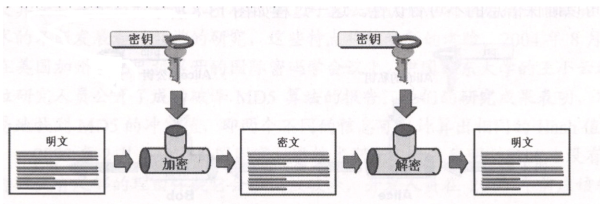



图 9-1　对称加密示意图

### 9.2.2 非对称加密、解密

非对称加密、解密是加密和解密使用不同的密码，有代表性的有RSA、DSA、ElGamal和ECDSA。非对称加密、解密的安全性是基于复杂数学难题，特点是运算复杂、速度慢，主要应用于金融、军事等重大机密系统，如图 9-2 所示。

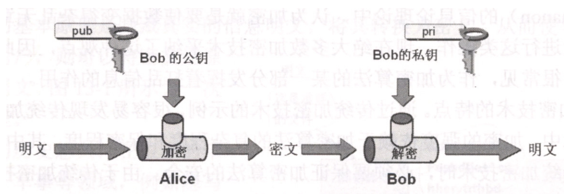



图 9-2　非对称加密示意图

为什么需要非对称加密和解密呢？假设你要发送一份非常机密的文件给你的朋友，而你朋友的电脑和外界通信的线路包括无线信号

都被监听了，那怎么办？

方案A：提前与你的朋友协商一个密码，然后使用对称加密算法加密文件，转送给你的朋友，你的朋友根据之前协商的密码解密文件。

使用方案A传输加密后的机密文件，即使监听者获得了文件，仍然无法解密。初步肯定方案A有效。但当情况变成你要给多个陌生人传输机密文件，而之前根本无法和每个陌生人协商密码，那怎么办呢？这个时候非对称加密就有用武之地了。

如果使用非对称加密算法来加密、解密机密文件，那么陌生人先要确定一个加密的密码，然后将加密密码传输给你，同时告诉你他使用哪种非对称加密算法，你用加密密码按照指定的非对称加密算法加密机密文件，将加密后的文件传输给陌生人，陌生人收到文件后使用另外一个密码解密文件。

这样即使监听者获得了加密密码和加密后的文件，也解不开机密文件，因为使用非对称加密算法加密的文件必须使用解密密码来解密，使用原来的加密密码来解密是无效的。

加密密码和解密密码是相对的，如果用加密密码加密，那么只有解密密码才能解密，如果用解密密码加密则只有加密密码能解密，所以它们被称为密码对，其中的一个可以在网络上发送、公布，叫做公钥，而另一个则只有密钥对的所有人才持有，叫做私钥，私钥不以任何形式传播。

对称加密、解密和非对称加密、解密主要有以下两个区别。

- 对称加密、解密使用分组加密、解密方法对一段明文按固定长度划分成多组，然后分别对每组使用可变长度的密码迭代编码，最后将每组编码后的密文重新组合在一起，解密时也一样，先对密文按固定长度划分成多组，然后使用密码解密每一组，最后重新组合明文。非对称加密、解密算法使用几乎不可逆的算法来加密、解密，加密和解密使用不同的密码。
- 对称加密、解密算法和非对称加密、解密算法都是公开的，对称加密、解密算法的安全性依赖于密码的强度和密码的保存，而非对称加密、解密算法只需公开一对密码中的其中一个，另外一个无须公开，所以非对称加密、解密算法的安全性比较大。

总体来说，非对称加密技术只要确保私钥的安全性即可，而公钥则公开，弥补对称加密技术密钥传递带来的失密问题。但由于非对称加密算法通常采用密集的数学计算，速度比对称加密要慢得多。实际应用中常将两者结合，用对称加密技术加密要传输的大量信息，而用非对称加密技术加密对称加密技术的共享密钥。

### 9.2.3 公钥结构的保密通信原理

要进行保密通信，发送方使用接收方的公钥对明文进行加密，接收方使用自己的私钥对密文进行解密。由于只有接收方才能对由自己的公钥加密的信息解密，因此可以实现保密通信，如图 9-3 所示。




图 9-3　公钥结构下的保密通信原理示意

### 9.2.4 公钥结构的鉴别通信原理

要进行鉴别通信，发送方使用自己的私钥对明文进行加密，接收方使用发送方的公钥对密文进行解密。接收方使用发送方的公钥进行解密，可以确信信息是由发送方加密的，因此可以鉴别发送方的身份，如图 9-4 所示。

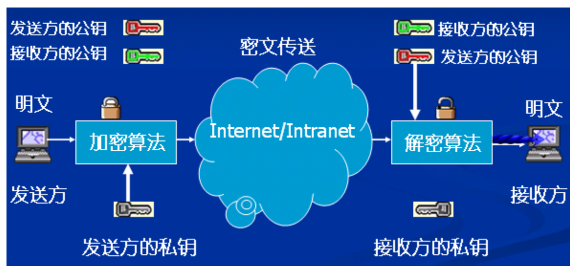



图 9-4 公钥结构下的鉴别通信原理示意

### 9.2.5 公钥结构的鉴别+保密通信原理

发送方先使用自己的私钥对明文进行加密，然后使用接收方的公钥进行加密。接收方先使用自己的私钥进行解密，然后使用发送方的公钥进行解密，这样就实现了鉴别和保密通信，如图 9-5 所示。

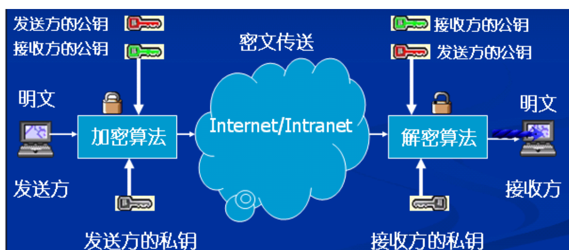



图 9-5 公钥结构的鉴别+保密通信的原理示意

## 9.3 应用一：使用 GnuPG 进行应用数据加密

随着网络与计算机技术的发展，数据存储与数据交换的安全性变得越来越重要，加密技术很早就用于数据存储和数据交换。GnuPG（GNU Privacy Guard）就是用来加密数据与制作证书的一套工具，其作用与PGP类似。但是PGP使用了许多专利算法，属于“臭名昭著”的美国加密出口限制之列。GnuPG是GPL软件，并且没有使用任何专利加密算法，所以使用起来有着更多的自由。

具体地说，GnuPG是实现安全通信和数据存储的一系列工具集，可以做加密数据和数字签名之用。在功能上，它和PGP是一样的。由于PGP使用了IDEA专利算法，所以使用PGP会有许可证的麻烦，但GnuPG并没有使用这个算法，所以对用户来说使用GnuPG没有任何限制。GnuPG使用非对称加密算法，安全程度比较高。所谓非对称加密算法，就是每一个用户都拥有一对密钥：公钥和私钥。其中，密钥由用户保存，公钥则由用户尽可能地散发给其他人，以便用户之间的通信。该软件可以从网站http://www.gnupg.org/上下载安装。

### 9.3.1 安装 GnuPG

很多开源系统都自带了GnuPG的软件安装包，用户可以在系统安装时选择安装，也可以以后安装。

通常在系统中会有两个已经安装的GnuPG软件包，一个是GnuPG 1.x，另一个是GnuPG 2.x。后者是GnuPG的最新稳定版本，兼容OpenPGP和S/MIME，并且不会和已经安装的 1.x系列有任何冲突。相比 1.x系列来说，由于有些新功能（支持S/MIME）的加入，所以其在运行时间和软件包

大小上都比1.x系列要大。但是从功能实现上来说，与1.x相差无几。为了解释和讨论的有效性和一致性，本章将采用GnuPG 1.x系列进行讲解说明。

## 9.3.2　GnuPG的基本命令

GnuPG支持的算法如下。

- 公钥：RSA、RSA-E、RSA-S、ELG-E、DSA。
- 对称加密：3DES、CAST5、BLOWFISH、AES、AES192、AES256、TWOFISH。
- 散列：MD5、SHA1、RIPEMD160、SHA256、SHA384、SHA512。
- 压缩：不压缩、ZIP、ZLIB、BZIP2。

其使用的基本语法为：gpg [选项] [文件名]

其实现的功能包括签名、检查、加密或解密，默认的操作依输入数据而定。

## 9.3.3　GnuPG的详细使用方法

### 1．生成密钥对

使用GnuPG之前必须生成密钥对（公钥和私钥），参数选项“--gen-key”可以生成密钥对。可按以下步骤进行操作，如图9-6和图9-7所示。

在图9-6中，读者首先需要注意有以下几个关键的步骤。

（1）GnuPG要求输入将生成的密钥的算法。GnuPG可以生成多种密钥对，这里有三种选择。DSA密钥是生成证书的最基本的密钥格式。ElGamal密钥对可以用来加密。第二种选择与第一种相似，但是仅仅生成DSA密钥对，第三种选择可以生成供签证和加密使用的ElGamal密钥对。对大多数用户来说，使用默认选择是非常方便的。

（2）选择密钥的长度。DSA密钥的长度在512～1024位之间，ElGamal密钥的长度则没有限制。生成一个很长的密钥既有优点也有缺点，长的密钥无疑安全性非常高，但是会导致加密的过程变得缓慢，另外，密钥过长，也会使证书的长度变大。默认密钥长度1024位已经够用了，确定了密钥的长度之后，就不能再改变它。

（3）需要指定这个密钥对的有效日期。如果选择了生成ElGamal或者 DSA密钥对，需要指定密钥对的失效日期。对于大多数用户来说，密钥对没有失效期限是可以的。虽然在密钥对产生以后，可以改变它的有效日期，但仍要谨慎选择该参数。因为公钥发送出去以后，很难再改变其他用户拥有的您的公钥。

```
[fedora@localhost ~]$ gpg --gen-key
gpg (GnuPG) 1.4.9; Copyright (C) 2008 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

请选择您要使用的密钥种类：
   (1) DSA 和 ElGamal (默认)
   (2) DSA (仅用于签名)
   (5) RSA (仅用于签名)
您的选择？ 1
DSA 密钥对会有 1024 位。
ELG-E 密钥长度应在 1024 位与 4096 位之间。
您想要用多大的密钥尺寸？(2048)1024
您所要求的密钥尺寸是 1024 位
请设定这把密钥的有效期限。
         0 = 密钥永不过期
      <n>  = 密钥在 n 天后过期
      <n>w = 密钥在 n 周后过期
      <n>m = 密钥在 n 月后过期
      <n>y = 密钥在 n 年后过期
密钥的有效期限是？(0) 0
密钥永远不会过期
以上正确吗？(y/n)y
```

图9-6　生成密钥对的前三个关键步骤

完成上述步骤后，用户还需要注意如图9-7所示的后续几个关键步骤。

（1）用户需要指定一个用户ID来标识选择的密钥，GnuPG可以根据用户的真实姓名、注释和E-mail地址产生一个用户ID。在图 9-7中，我们使用姓名（liyang），电子邮件地址（liyang@tsinghua.com）和注释（liyang@tsinghua），

并设定了密钥的密码，来完成该步骤。这个口令的目的是用来加密用户的私钥，这样，即使有人偷走了用户的私钥，没有这个口令，也无法使用，这个口令的长度没有限制，但是，正如我们所知道的，一个短的口令是很容易被破解的。同样，如果用户的口令是一个单词，也很容易被破解。

（2）在产生密钥的过程中，GnuPG需要得到一些随机的数字。这些随机的数字可以从用户系统当前的状态中得到，所以这时候，用户可以随机敲一下键盘或者移动鼠标，来产生高质量的随机数。图 9-7 中系统就明显地要求用户尽量多产生一些随机数字来生成密钥，以保证质量。

完成了上述 5 个关键步骤后，如果系统显示如图 9-8 所示的成功界面，则生成密钥成功，否则用户需要再重复以上 5 个步骤。

```
密钥的有效期限是？(0) 0
密钥永远不会过期
以上正确吗？(y/n)y

您需要一个用户标识来辨识您的密钥；本软件会用真实姓名、注释和电子邮件地址组合
成用户标识，如下所示：
    “Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>”

真实姓名：liyang
电子邮件地址：liyang@tsinghua.com
注释：liyang@tsinghua
您选定了这个用户标识：
    “liyang (liyang@tsinghua) <liyang@tsinghua.com>”

更改姓名(N)、注释(C)、电子邮件地址(E)或确定(O)/退出(Q)？o
您需要一个密码来保护您的私钥。

我们需要生成大量的随机字节。这个时候您可以多做些琐事(像是敲打键盘、移动
鼠标、读写硬盘之类的)，这会让随机数字发生器有更好的机会获得足够的熵数。
++++++++++++++++++++++++..+++++.++++++++++.+++++++++++++++++++++++++++++++++++++++++++++++++++.++
++++++++.++++++++++..++++++++++++++++++++.+++++.............................................
>+++++..<+++++..........>+++++..<+++++...............................................>+++
++...................+++++
```

图 9-7　生成密钥的后续 2 个关键步骤

```
我们需要生成大量的随机字节。这个时候您可以多做些琐事(像是敲打键盘、移动
鼠标、读写硬盘之类的)，这会让随机数字发生器有更好的机会获得足够的熵数。
.+++++++++++++++++++++++++++++++++++.+++++++++++++++++++++++++..+++++++++++++++.+++++++++++++++++
+++++++++++++++++++++++++++.++++++++++................>.+++++....................................
..................................................>+++++.......<+++++.>.+++++..................
.......+++++^^^

^^^^^^^^
gpg: /home/fedora/.gnupg/trustdb.gpg：建立了信任度数据库
gpg: 密钥 4B4070F9 被标记为绝对信任
公钥和私钥已经生成并经签名。

gpg: 正在检查信任度数据库
gpg: 需要 3 份勉强信任和 1 份完全信任，PGP 信任模型
gpg: 深度：0 有效性：  1 已签名：  0 信任度：0-, 0q, 0n, 0m, 0f, 1u
pub   1024D/4B4070F9 2009-10-07
密钥指纹 = 7F88 F1DB 9E62 247C B3DE  9A5C E334 78FD 4B40 70F9
uid                  liyang (liyang@tsinghua) <liyang@tsinghua.com>
```

图 9-8　生成密钥成功

### 2．为密钥建立吊销证书

当用户的密钥对生成之后，用户应该立即做一个公钥回收证书，如果忘记了私钥的口令或者私钥丢失或者被盗窃，用户可以发布这个证书来声明以前的公钥不再有效。生成回收证书的选项是“--gen-revoke”。具体使用的命令是：

```
# gpg --output revoke.asc --gen-revoke mykey
```

其中mykey 参数是可以表示的密钥标识，产生的回收证书放在revoke.asc文件中，一旦回收证书被发放，以前的证书就不能再被其他用户访问，因此以前的公钥也就失效了。具体的过程如图 9-9～图 9-11 所示。在该过程中我们为用户liyang的密钥建立了一份吊销证书，在建立过程中需要依次输入吊销理由和为密钥设定的密码才能成功建立吊销证书。

```
[fedora@localhost ~]$ gpg --output liyang_revoke.asc --gen-revoke liyang

sec  1024D/4B4070F9 2009-10-07 liyang (liyang@tsinghua) <liyang@tsinghua.com>

要为这把密钥建立一份吊销证书吗？(y/N)y
请选择吊销的原因：
  0 = 未指定原因
  1 = 密钥已泄漏
  2 = 密钥被替换
  3 = 密钥不再使用
  Q = 取消
(也许您会想要在这里选择 1)
您的决定是什么？1
请输入描述(可选)；以空白行结束：
>
吊销原因：密钥已泄漏
(不给定描述)
这样可以吗？ (y/N)y
```

图 9-9　输入吊销理由

```
[fedora@localhost ~]$ gpg --output liyang_revoke.asc --gen-revoke liyang

sec  1024D/4B4070F9 2009-10-07 liyang (liyang@tsinghua) <liyang@tsinghua.com>

要为这把密钥建立一份吊销证书吗？(y/N)y
请选择吊销的原因：
  0 = 未指定原因
  1 = 密钥已泄漏
  2 = 密钥被替换
  3 = 密钥不再使用
  Q = 取消
(也许您会想要在这里选择 1)
您的决定是什么？1
请输入描述(可选)；以空白行结束：
>
吊销原因：密钥已泄漏
(不给定描述)
这样可以吗？ (y/N)y

您需要输入密码，才能解开这个用户的私钥：“liyang (liyang@tsinghua) <liyang@tsinghua.com>”
1024 位的 DSA 密钥，钥匙号 4B4070F9，建立于 2009-10-07

请输入密码：
```

图 9-10　输入为密钥建立的密码

```
[fedora@localhost ~]$ gpg --output liyang_revoke.asc --gen-revoke liyang

sec  1024D/4B4070F9 2009-10-07 liyang (liyang@tsinghua) <liyang@tsinghua.com>

要为这把密钥建立一份吊销证书吗？(y/N)y
请选择吊销的原因：
  0 = 未指定原因
  1 = 密钥已泄漏
  2 = 密钥被替换
  3 = 密钥不再使用
  Q = 取消
(也许您会想要在这里选择 1)
您的决定是什么？1
请输入描述(可选)；以空白行结束：
>
吊销原因：密钥已泄漏
(不给定描述)
这样可以吗？ (y/N)y

您需要输入密码，才能解开这个用户的私钥：“liyang (liyang@tsinghua) <liyang@tsinghua.com>”
1024 位的 DSA 密钥，钥匙号 4B4070F9，建立于 2009-10-07

已强行使用 ASCII 封装过的输出。
已建立吊销证书。
```

图 9-11　成功建立吊销证书

### 3．显示密钥列表

完成上述操作后可以使用--list-keys 选项列出我们生成的密钥，如图9-12所示。

```
[fedora@localhost ~]$ gpg --list-keys
/home/fedora/.gnupg/pubring.gpg
-------------------------------
pub   1024D/4B4070F9 2009-10-07
uid                  liyang (liyang@tsinghua) <liyang@tsinghua.com>
sub   1024g/433F132B 2009-10-07
```

图9-12　密钥列表

### 4．输出公钥

用户可以输出自己的公钥供其主页使用，也可以把它放在密钥服务器上，当然，还可以使用其他的途径。在使用此公钥之前用户首先要导出它。选项--export可以实现该功能，在使用这个选项时，还必须使用附加的选项指明用户要输出的公钥。

下面的命令表示以二进制格式输出公钥：

```
# gpg --output pubring.gpg --export samsunglinux@minigui.org
```

下面的命令表示以ASCII字符格式输出：

```
#gpg --output pubring.gpg --export--armor> liyang_public-key.asc
```

### 5．导入公钥

用户可以把从第三方的公钥数据库中得到的公钥导入自己的私有数据库，在与他人进行通信时使用。命令如下：

```
#gpg --import < filename >
```

其中，参数filename为公钥文件。

图9-13给出了将用户liyang的公钥导入到用户samsunglinux自己的私有数据库的例子。

```
[samsunglinux@localhost ~]$ gpg --import pubring.gpg
gpg: /home/samsunglinux/.gnupg/trustdb.gpg：建立了信任度数据库
gpg: 密钥 4B4070F9：公钥“liyang (liyang@tsinghua) <liyang@tsinghua.com>”已导入
gpg: 合计被处理的数量：1
gpg:           已导入：1
[samsunglinux@localhost ~]$ ll
总计 4
-rw-r--r-- 1 root root 922 10-07 12:13 pubring.gpg
[samsunglinux@localhost ~]$ ls -a
.   .bash_logout   .bashrc   .gnupg    pubring.gpg
..  .bash_profile  .gnome2   .mozilla  .zshrc
[samsunglinux@localhost ~]$ cd .gnupg
[samsunglinux@localhost .gnupg]$ ll
总计 20
-rw------- 1 samsunglinux samsunglinux 9154 10-07 12:12 gpg.conf
-rw------- 1 samsunglinux samsunglinux  922 10-07 12:14 pubring.gpg
-rw------- 1 samsunglinux samsunglinux    0 10-07 12:12 pubring.gpg~
-rw------- 1 samsunglinux samsunglinux    0 10-07 12:12 secring.gpg
-rw------- 1 samsunglinux samsunglinux 1200 10-07 12:14 trustdb.gpg
```

图9-13　导入公钥示例

### 6．确认密钥

导入密钥以后，使用数字签名来验证此证书是否合法。查看数字签名使用--fingerprint选项。其命令如下：

```
#gpg --fingerprint < UID >
```

其中，UID为用户要验证的公钥。图9-14给出了验证证书的例子。

```
[samsunglinux@localhost ~]$ gpg --fingerprint liyang
pub   1024D/4B4070F9 2009-10-07
密钥指纹 = 7F88 F1DB 9E62 247C B3DE  9A5C E334 78FD 4B40 70F9
uid                  liyang (liyang@tsinghua) <liyang@tsinghua.com>
sub   1024g/433F132B 2009-10-07
```

图9-14　确认密钥示意

### 7．密钥签名

导入密钥之后，可以使用--sign-key 选项进行签名，签名的目的是证明用户完全信任这个证书的合法性。其命令格式为：

```
# gpg --sign-key < UID >
```

其中，UID 是要签名的公钥。

### 8．检查签名

用户可以使用--check-sigs 选项来检查在上面对密钥所作的签名。其命令格式为：

```
# gpg --check-sigs < UID >
```

这个选项可以列出此密钥文件的所有的签名。

#### 9．加密和解密

使用GnuPG加密和解密一个文件非常容易，如果用户要给对方发送一个加密文件，可以使用对方的公钥加密这个文件，并且这个文件也只有对方使用自己的密钥才可以解密查看。

加密一个文件可以使用下面的指令：

```
#gpg -r < UID > --encrypt < file >
```

其中，UID是对方的公钥，file为要加密的文件。

相应的，如果用户要解开一个其他用户发给自己的文件，可以使用下面的指令：

```
#gpg -d < file >
```

其中，file是要解密的文件。解密过程中，GnuPG会提示用户输入使用密钥所需的口令，也就是在产生私钥时用户所输入的口令，否则，该文件将无法正常解密并被用户使用。

图 9-15 和图 9-16 分别显示了用户samsung Linux对文件gpg.conf进行加密传输，用户liyang对该加密文件gpg.conf.gpg进行解密的过程。

```
[samsunglinux@localhost .gnupg]$ gpg -r liyang --encrypt gpg.conf
gpg: 433F132B：没有证据表明这把密钥真的属于它所声称的持有者

pub  1024g/433F132B 2009-10-07 liyang (liyang@tsinghua) <liyang@tsingh
a.com>
 主钥指纹：  7F88 F1DB 9E62 247C B3DE  9A5C E334 78FD 4B40 70F9
 子钥指纹：  2248 ECA7 A11B 0F3E 127F  3971 677D F588 433F 132B

这把密钥并不一定属于用户标识声称的那个人。如果您真的知道自
己在做什么，您可以在下一个问题回答 yes。

无论如何还是使用这把密钥吗？(y/N)y
[samsunglinux@localhost .gnupg]$ ll
总计 32
-rw------- 1 samsunglinux samsunglinux 9154 10-07 12:12 gpg.conf
-rw-rw-r-- 1 samsunglinux samsunglinux 4172 10-07 13:15 gpg.conf.gpg
-rw------- 1 samsunglinux samsunglinux  922 10-07 12:14 pubring.gpg
-rw------- 1 samsunglinux samsunglinux    0 10-07 12:12 pubring.gpg~
-rw------- 1 samsunglinux samsunglinux  600 10-07 13:15 random_seed
```

图 9-15　用户 samsunglinux 对文件 gpg.conf 进行加密

```
[fedora@localhost .gnupg]$ gpg -d gpg.conf.gpg

您需要输入密码，才能解开这个用户的私钥：“liyang (liyang@tsinghua) <liyang@tsin
ghua.com>”
1024 位的 ELG-E 密钥，钥匙号 433F132B，建立于 2009-10-07 (主钥匙号 4B4070F9)

gpg: 由 1024 位的 ELG-E 密钥加密，钥匙号为 433F132B、生成于 2009-10-07
      “liyang (liyang@tsinghua) <liyang@tsinghua.com>”
# Options for GnuPG
# Copyright 1998, 1999, 2000, 2001, 2002, 2003 Free Software Foundation, Inc.
#
# This file is free software; as a special exception the author gives
# unlimited permission to copy and/or distribute it, with or without
# modifications, as long as this notice is preserved.
#
# This file is distributed in the hope that it will be useful, but
# WITHOUT ANY WARRANTY, to the extent permitted by law; without even the
# implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

图 9-16　用户 liyang 对文件进行解密并浏览（白色部分）

### 9.3.4　GnuPG 使用实例

下面详细介绍如何使用GnuPG收发数据，主要会涉及到数据交换的两种方式。

- 数字签名传输（signed data）：发送者使用私钥对数据加密，接收者使用公钥对数据解密。
- 数据加密传输（encrypted data）：发送者使用公钥对数据加密，接收者使用私钥对数据解密。

#### 1. 数字签名传输

发送者使用私钥对数据进行签名，接收者拥有发送者的公钥，对之信任并使用它验证接收数据的完整性。对数据进行签名的最简单的方法是使用clearsign命令，这将使GnuPG创建一个易读的签名，很适于发送E-mail。具体命令及执行情况如下：

```
#gpg --clearsign mymessage.txt
```

输入密码后，将生成一个扩展名为.asc的新文件，这里就是transmit.txt.asc。这个文件包含了transmit.txt文件的原始内容以及签名信息，如图 9-17 所示。

```
[fedora@localhost .gnupg]$ cat transmit.txt.asc
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

hi,this is a test for GnuPG encrypted file transmitted!

One world, One dream!
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.9 (GNU/Linux)

iEYEARECAAYFAkrMJSoACgkQ4zR4/UtAcPnpSgCgrNEOnHodvxSfZaBaeDjvtGAs
FPUAn3DHai3bdnUAqbGzxqOQdPcepbVp
=J2f1
-----END PGP SIGNATURE-----
```

图 9-17　文件 transmit.txt.asc 的内容

当接收者收到包含上述签名的信息或文件时，可以使用发送者的公钥来验证信息的完整性，具体命令及执行情况如下，参见图 9-18 所示。

```
[fedora@localhost .gnupg]$ gpg --verify transmit.txt.asc
gpg: 于 2009年10月07日 星期三 13时20分42秒 CST 创建的签名，使用 DSA，钥
4070F9
gpg: 完好的签名，来自于“liyang (liyang@tsinghua) <liyang@tsinghua.com>”
```

图 9-18　验证数字签名的完整性

```
#gpg --verify transmit.txt.asc
```

2．数据加密传输

这种传输方式的目的是为了只让个别人看到发送信息者使用其公钥对文件或数据进行加密，接收者使用发送者的私钥对接收数据进行解密。加密命令包含两个部分，一部分指定接收者的E-mail，另一部分指定要加密的文件。这里只给出一个加密后的transmit.txt文件的示例，如图 9-19 所示。在网络中传输时，即使该文件被黑客或者其他用户截获，但没有私钥，该用户也无法对该文件进行识别，因而具有很高的安全性。

```
[fedora@localhost .gnupg]$ cat transmit.txt.gpg
```

图 9-19　加密后的 transmit.txt 文件

另外，值得一提的是：通过以上方式被加密的信息也可以被签名，方法是在上述命令中再加上一个-s参数。如下命令：

```
#gpg -r < UID > --encrypt -s< file >
```

那么，在使用-d选项来解密该文件时，将会出现如图 9-20 所示的情况，该图的白色部分给出了对签名的检验情况。

```
[fedora@localhost .gnupg]$ gpg -d transmit.txt.gpg

您需要输入密码，才能解开这个用户的私钥：“liyang (liyang@tsinghua) <liya
ng@tsinghua.com>”
1024 位的 ELG-E 密钥，钥匙号 433F132B，建立于 2009-10-07 (主钥匙号 4B40
70F9)

gpg: 由 1024 位的 ELG-E 密钥加密，钥匙号为 433F132B、生成于 2009-10-07
      “liyang (liyang@tsinghua) <liyang@tsinghua.com>”
hi,this is a test for GnuPG encrypted file transmitted!

One world, One dream!
gpg: 于 2009年10月07日 星期三 13时33分02秒 CST 创建的签名，使用 DSA，钥
匙号 4B4070F9
gpg: 完好的签名，来自于“liyang (liyang@tsinghua) <liyang@tsinghua.com>”
```

图 9-20　解密过程中检验数字签名

### 9.3.5　GnuPG 使用中的注意事项

在使用GnuPG的过程中，需要注意以下几个问题。

- 需要根据实际应用来确定生成密钥的算法、密钥的长度以及密钥的有效期限。
- 需要用户通过交互移动鼠标、键盘来保证生成的密钥对的随机性；否则，极有可能被黑客破解。
- 公钥的安全性问题是 GnuPG 安全的核心，一个成熟的加密体系必然要有一个成熟的密钥管理机制配套。公钥体制的提出就是为了解决传统加密体系的密钥分配难保密的缺点。比如网络黑客常用的手段之一就是“监听”，如果密钥是通过网络传送就太危险了。对 GnuPG 来说公钥本来就要公开，因此没有防监听的问题。但公钥的发布中仍然存在安全性问题，例如公钥被篡改，这可能是公钥密码体系中最大的漏洞，因为大多数新手不能很快发现这一点。你必须确信自己拿到的公钥属于它看上去属于的那个人。
- 私钥的保密也是决定性的。相对公钥而言，私钥不存在被篡改的问题，但存在泄露的问题。GnuPG 的办法是让用户为随机生成的 RSA 私钥指定一个口令。只有通过给出的口令才能将私钥释放出来使用，用口令加密私钥的方法的保密程度和 GnuPG 本身是一样的。所以私钥的安全性问题实际上首先是对用户口令的保密。当然私钥文件本身失密也很危险，因为破译者所需要的只是用穷举法试探出你的口令，虽说很困难，但毕竟是损失了一层安全性。在这里只用简单地记住一点，要像任何隐私一样保存你的私

钥，不要让任何人有机会接触到它。

- 在实际的使用过程中，用户可以将 GnuPG 软件灵活地运用到网络数据传输，包括电子邮件发送，FTP 文件传送等各个应用领域。

## 9.4 应用二：使用 SSH 加密数据传输通道

SSH的英文全称是Secure SHell。通过使用SSH，用户可以把所有传输的数据进行加密，这样，“中间人”这种攻击方式就不可能实现了，而且也能够防止DNS和IP欺骗。还有一个额外的好处就是传输的数据是经过压缩的，所以可以加快传输的速度。SSH有很多功能，它既可以代替Telnet，又可以为FTP、POP，甚至PPP提供一个安全的“通道”。

SSH协议是建立在应用层和传输层基础上的安全协议，主要由以下三部分组成，共同实现SSH的安全保密机制。

- 传输层协议。该协议提供诸如认证、信任和完整性检验等安全措施，此外还可以任意地提供数据压缩功能。通常情况下，这些传输层协议都建立在面向连接的 TCP 数据流之上。
- 用户认证协议层。用来实现服务器和客户端用户之间的身份认证，其运行在传输层协议之上。
- 连接协议层。分配多个加密通道至一些逻辑通道上，其运行在用户认证层协议之上。

SSH是由客户端和服务端的软件组成的，有两个不兼容的版本，分别是 1.x和 2.x。用SSH 2.x客户程序是不能连接到SSH 1.x的服务程序上去的。OpenSSH 2.x同时支持SSH 1.x和 2.x。

从客户端来看，SSH提供两种级别的安全验证。

第一种级别（基于口令的安全验证）：只要用户知道自己的账号和口令，就可以登录到远程主机。所有传输的数据都会被加密，但是不能保证用户正在连接的服务器就是用户想连接的服务器。可能会有别的服务器在冒充真正的服务器，也就是受到“中间人”这种方式的攻击。

第二种级别（基于密钥的安全验证）：需要依靠密钥，也就是用户必须为自己创建一对密钥，并把公用密钥放在需要访问的服务器上。如果用户要连接到SSH服务器上，客户端软件就会向服务器发出请求，请求用用户的密钥进行安全验证。服务器收到请求之后，先在用户的该服务器的主目录下寻找用户的公用密钥，然后把它和用户发送过来的公用密钥进行比较。如果两个密钥一致，服务器就用公用密钥加密“质询”（challenge）并把它发送给客户端软件。客户端软件收到“质询”之后就可以用用户的私人密钥解密再把它发送给服务器。

用这种方式，必须知道自己密钥的口令。但是，与第一种级别相比，第二种级别不需要在网络上传送口令。第二种级别不仅加密所有传送的数据，而且“中间人”这种攻击方式也是不可能的（因为它没有用户的私人密钥）。

### 9.4.1 安装最新版本的 OpenSSH

要安装最新版本的OpenSSH，可以从网站http://www.openssh.com/上下载其发布的最新版本openssh-5.2p1.tar.gz，然后按照如下命令进行安装即可。

（1）解压缩源码包：

```
#tar xzvf openssh-5.2p1.tar.gz
```

（2）预编译，生成makefile文件：

```
#cd openssh-5.2p1
#./configure
```

（3）安装：

```
#make
#make install
```

安装完成之后，可以使用下述命令（Red Hat和Fedora系列）进行启动：

```
#service sshd start
```

或者使用以下命令（Ubuntu系列）进行启动：

```
#/etc/initd/sshd start
```

### 9.4.2 配置OpenSSH

#### 1．与SSH有关的配置文件

OpenSSH的设置文件和主要文件存放在/etc/ssh/目录中，主要包括以下文件。

- /etc/ssh/sshd_config：sshd 服务器的设置文件。
- /etc/ssh/ssh_config：ssh 客户机的设置文件。
- /etc/ssh/ssh_host_key：SSH1 用的 RSA 私钥。
- /etc/ssh/ssh_host_key.pub：SSH1 用的 RSA 公钥。
- /etc/ssh/ssh_host_rsa_key：SSH2 用的 RSA 私钥。
- /etc/ssh/ssh_host_rsa_key.pub：SSH2 用的 RSA 公钥。
- /etc/ssh/ssh_host_dsa_key：SSH2 用的 DSA 私钥。
- /etc/ssh/ssh_host_dsa_key.pub：SSH2 用的 DSA 公钥。

#### 2．配置/etc/ssh/ssh_config文件

/etc/ssh/ssh_config文件是OpenSSH系统范围的配置文件，允许用户通过设置不同的选项来改变客户端程序的运行方式。该文件的每一行包含“关键词－值”的匹配，其中“关键词”是忽略大小写的。下面列出最重要的关键词，用man命令查看帮助页（ssh（1））可以得到详细的列表。

配置需要编辑ssh_config文件（vi /etc/ssh/ssh_config），添加或改变下面的参数：

```
# Site-wide defaults for various options
Host *
ForwardAgent no
ForwardX11 no
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication yes
FallBackToRsh no
UseRsh no
BatchMode no
```

```
CheckHostIP yes
StrictHostKeyChecking no
IdentityFile ~/.ssh/identity
Port 22
Cipher blowfish
EscapeChar ~
```

下面逐行说明上面的选项设置。

- Host *：只对能够匹配后面字串的计算机有效。“*”表示所有的计算机。
- ForwardAgent no：ForwardAgent 设置连接是否经过验证代理（如果存在）转发给远程计算机。
- ForwardX11 no：设置 X11 连接是否被自动重定向到安全的通道和显示集（DISPLAY set）。
- RhostsAuthentication no：是否用基于 rhosts 的安全验证。
- RhostsRSAAuthentication no：是否用 RSA 算法的基于 rhosts 的安全验证。
- RSAAuthentication yes：是否用 RSA 算法进行安全验证。
- PasswordAuthentication yes：是否用口令验证。
- FallBackToRsh no：如果用 SSH 连接出现错误是否自动使用 RSH。
- UseRsh no：是否在这台计算机上使用 rlogin/rsh。
- BatchMode no：如果设为 yes，passphrase/password（交互式输入口令）的提示将被禁止。当不能交互式输入口令的时候，该选项对脚本文件和批处理任务十分有用。
- CheckHostIP yes：设置 SSH 是否查看连接到服务器的主机的 IP 地址以防止 DNS 欺骗。建议设置为 yes。
- StrictHostKeyChecking no：如果设置为 yes，SSH 就不会自动把计算机的密钥加入 $HOME/.ssh/known_hosts 文件，并且一旦计算机的密钥发生了变化，就拒绝连接。
- IdentityFile～/.ssh/identity：设置从哪个文件读取用户的 RSA 安全验证标识。
- Port 22：设置连接到远程主机的端口。
- Cipher blowfish：设置加密用的密码。
- EscapeChar～：设置 escape 字符。

假定用户在www.super.com上有一个名为“baby”的账号。而且要把ssh-agent和ssh- add结合起来使用，并且使用数据压缩来加快传输速度。因为主机名太长了，想使用super作为www.super.com的简称。那么，配置文件可以如下编写：

```
Host *super
HostName www.super.com
User baby
ForwardAgent yes
Compression yes
# Be paranoid by default
Host *
ForwardAgent no
ForwardX11 no
```

```
FallBackToRsh no
```

当用户输入ssh super之后，SSH会自动地从配置文件中找到主机的全名，使用用户名登录并且用ssh-agent管理的密钥进行安全验证。

### 3．配置/etc/ssh/sshd_config 文件

/etc/ssh/sshd_config是OpenSSH的配置文件，允许设置选项改变这个daemon的运行。这个文件的每一行包含“关键词一值”的匹配，其中“关键词”是忽略大小写的。下面列出来的是最重要的关键词，用man命令查看帮助页（sshd（8））可以得到详细的列表。

编辑sshd_config文件（vi /etc/ssh/sshd_config），添加或改变下面的参数：

```
# This is ssh server systemwide configuration file.
Port 22
ListenAddress 192.169.1.1
HostKey /etc/ssh/ssh_host_key
ServerKeyBits 1024
LoginGraceTime 600
KeyRegenerationInterval 3600
PermitRootLogin no
IgnoreRhosts yes
IgnoreUserKnownHosts yes
StrictModes yes
X11Forwarding no
PrintMotd yes
SyslogFacility AUTH
LogLevel INFO
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication yes
PermitEmptyPasswords no
AllowUsers admin
```

下面逐行说明上面选项的设置。

- Port 22：Port 设置 sshd 监听的端口号。
- ListenAddress 192.169.1.1：ListenAddress 设置 sshd 服务器绑定的 IP 地址。
- HostKey /etc/ssh/ssh_host_key：HostKey 设置包含计算机私人密钥的文件。
- ServerKeyBits 1024：ServerKeyBits 定义服务器密钥的位数。
- LoginGraceTime 600：LoginGraceTime 设置如果用户不能成功登录，在切断连接之前服务器需要等待的时间（以秒为单位）。
- KeyRegenerationInterval 3600：KeyRegenerationInterval 设置在多少秒之后自动重新生成服务器的密钥（如果使用密钥）。重新生成密钥是为了防止使用盗用的密钥解密被截

获的信息。

- PermitRootLogin no：PermitRootLogin 设置 root 能不能用 SSH 登录。该选项一定不要设成 yes。
- IgnoreRhosts yes：IgnoreRhosts 设置验证的时候是否使用 rhosts 和 shosts 文件。
- IgnoreUserKnownHosts yes：IgnoreUserKnownHosts 设置 ssh daemon 是否在进行 RhostsRSAAuthentication 安全验证的时候忽略用户的$HOME/.ssh/known_hosts。
- StrictModes yes：StrictModes 设置 SSH 在接收登录请求之前是否检查用户的目录和 rhosts 文件的权限和所有权。这通常是必要的，因为新手经常会把自己的目录和文件设成任何人都有写权限。
- X11Forwarding no：X11Forwarding 设置是否允许 X11 转发。
- PrintMotd yes：PrintMotd 设置 sshd 是否在用户登录的时候显示/etc/motd 中的信息。
- SyslogFacility AUTH：SyslogFacility 设置在记录来自 sshd 的消息的时候，是否给出 facility code。
- LogLevel INFO：LogLevel 设置记录 sshd 日志消息的层次。INFO 是一个好的选择。查看 sshd 的 man 帮助页，可以获取更多的信息。
- RhostsAuthentication no：RhostsAuthentication 设置只用 rhosts 或/etc/hosts.equiv 进行安全验证是否已经足够了。
- RhostsRSAAuthentication no：RhostsRSA 设置是否允许使用 rhosts 或/etc/hosts.equiv 加上 RSA 进行安全验证。
- RSAAuthentication yes：RSAAuthentication 设置是否允许只有 RSA 安全验证。
- PasswordAuthentication yes：PasswordAuthentication 设置是否允许口令验证。
- PermitEmptyPasswords no：PermitEmptyPasswords 设置是否允许用口令为空的账号登录。
- AllowUsers admin：AllowUsers 的后面可以跟任意数量的用户名的匹配串（patterns）或 user@host 这样的匹配串，这些字符串用空格隔开。主机名可以是 DNS 名或 IP 地址。

### 9.4.3 SSH 的密钥管理

SSH的密钥管理主要包括两个方面：生成公钥/私钥对以及公钥的分发。下面针对这两个密钥管理工作分别进行介绍。

#### 1．生成用户自己的密钥对

生成并分发用户自己的密钥有以下两个好处。

- 可以防止“中间人”这种攻击方式。
- 可以只用一个口令就登录到所有用户想登录的服务器上。

用下面的命令可以生成密钥：

```
#ssh-keygen
```

如果远程主机使用的是SSH 2.x就要使用下面的命令：

```
#ssh-keygen -d
```

在同一台主机上同时有SSH1 和SSH2 的密钥是没有问题的，因为密钥是保存为不同的文件的。

ssh-keygen命令运行之后会显示下面的信息：

```
Generating RSA keys:
```

```
Key generation complete.
Enter file in which to save the key (/home/[user]/.ssh/identity):
//此时按下回车键就行了
Created directory '/home/[user]/.ssh'.
Enter passphrase (empty for no passphrase):  //输入的口令不会显示在屏幕上
//重新输入一遍口令，如果忘记了口令就只能重新生成一次密钥了
Enter same passphrase again:
//保存用户的私人密钥和公用密钥
Your identification has been saved in /home/[user]/.ssh/identity.
Your public key has been saved in /home/[user]/.ssh/identity.pub.
The  key  fingerprint  is:  2a:dc:71:2f:27:84:a2:e4:a1:1e:a9:63:e2:fa:a5:89
[user]@[local machine]
```

ssh-keygen -d做的是几乎同样的事，但需把一对密钥存为（默认情况下）/home/[user]/.ssh/id_dsa（私人密钥）和/home/[user]/.ssh/id_dsa.pub（公用密钥）。

现在用户拥有一对密钥：公用密钥要分发到所有用户想用SSH登录的远程主机上去；私人密钥要好好地保管，防止别人知道私人密钥。用ls-l ~/.ssh/identity或ls-l ~/.ssh/id_dsa所显示的文件的访问权限必须是“-rw-------”。

**2．分发公用密钥**

在每一个用户需要用SSH连接的远程服务器上，用户要在自己的主目录下创建一个.ssh的子目录，把用户的公用密钥identity.pub拷贝到这个目录下并把它重命名为“authorized_ keys”。然后执行：

```
chmod 644 .ssh/authorized_keys
```

这一步是必不可少的。如果除了用户之外别人对authorized_keys文件也有写的权限，SSH就不会工作。

如果用户想从不同的计算机登录到远程主机，authorized_keys文件也可以有多个公用密钥。在这种情况下，必须在新的计算机上重新生成一对密钥，然后把生成的identify.pub文件拷贝并粘贴到远程主机的authorized_keys文件中。当然在新的计算机上用户必须有一个账号，而且密钥是用口令保护的。有一点很重要，就是当用户取消了该账号之后，别忘了把这一对密钥删掉。

### 9.4.4　使用scp命令远程拷贝文件

如前面所介绍的测试SSH服务器的功能步骤所示，SSH提供了一些命令和shell用来登录远程服务器，在默认情况下其并不允许用户拷贝文件。但为了方便用户使用，它还是提供了一个scp命令，用户可以使用该命令来进行文件的远程拷贝工作。

假定用户想把本地计算机当前目录下的一个名为“share”的文件拷贝到远程服务器www.remote.com上用户的家目录下。而且用户在远程服务器上的账号名为“super”。可以用这个命令：

```
scp share super@www.foobar.com:.
```

把文件拷贝回来用这个命令：

```
scp super@www.remote.com:share.
```

scp调用SSH进行登录，然后拷贝文件，最后调用SSH关闭这个连接。

如果在用户的~/.ssh/config文件中已经为www.foobar.com做了如下的配置：

```
Host *fbc
HostName www.remote.com
User super
ForwardAgent yes
```

那么用户就可以用fbc来代替bilbo@www.foobar.com，命令就简化为scp dumb fbc:.。

scp假定用户在远程主机上的家目录为用户的工作目录。如果用户使用相对目录就要相对于根目录。

用scp命令的-r参数允许递归地拷贝目录。scp也可以在两个不同的远程主机之间拷贝文件。

在使用过程中，有时候用户可能会试图进行如下操作：用SSH登录到www.remote.com上之后，输入命令“scp [local machine]:share .”，想用其把本地的share文件拷贝到用户当前登录的远程服务器上。这时候用户会看到下面的出错信息：

```
ssh: secure connection to [local machine] refused
```

之所以会出现这样的出错信息是因为用户运行的是远程的scp命令，该命令试图登录到在用户本地计算机上运行的SSH服务程序，而这样做是不允许的，除非用户的本地计算机也运行着SSH服务程序。

### 9.4.5 使用 SSH 设置“加密通道”

SSH的“加密通道”是通过“端口转发”来实现的。用户可以在本地端口（没有使用过的）和远程服务器上运行的某个服务的端口之间建立“加密通道”。所有对本地端口的请求都被SSH加密并且转发到远程服务器的端口。当然只有远程服务器上运行SSH服务器软件的时候“加密通道”才能工作。可以用下面的命令检查一些远程服务器是否运行SSH服务：

```
telnet [远程主机的名字全称] 22
```

如果收到这样的出错信息：

```
telnet: Unable to connect to remote host: Connection refused
```

就说明远程服务器上没有运行SSH服务软件。

端口转发使用以下命令语法：

```
ssh -f [远程主机上的用户名] -L [本地端口号]:[远程主机的名字全称]:[远程端口] [命令]
```

不仅可以转发多个端口，而且可以在~/.ssh/config文件中用LocalForward设置经常使用的一些转发端口。

#### 1. 为 POP 加上“加密通道”

可以使用POP协议从服务器上获取E-mail。为POP加上“加密通道”可以防止POP的密码被网络监听器（sniffer、TCPDUMP等软件）监听到。还有一个好处就是SSH的压缩方式可以让邮件传输得更快。

假定用户在pop.foobar.com上有一个POP账号，用户的用户名是“bilbo”，用户的POP口令是“topsecret”。用来建立SSH“加密通道”的命令是：

```
ssh -f -C bilbo@pop.foobar.com -L 1234:pop.foobar.com:110 sleep 5
```

（如果要测试，可以把sleep值加到 500）。运行这个命令之后会提示用户输入POP口令：

```
bilbo@pop.foobar.com's password:
```

输入口令之后就可以用telnet连接到本地的转发端口了。

```
telnet localhost 1234
```

用户会收到远程Mall服务器的READY消息。

当然，这个方法要求用户手工输入所有的POP命令，这是很不方便的。可以用Fetchmail（参考how to configure Fetchmail）。Secure POP via SSH mini-HOWTO、man fetchmail和在/usr/doc/fetchmail-[…]目录下的Fetchmail的FAQ都提供了一些具体的例子。

#### 2．为 X 加上“加密通道”

如果用户打算在本地计算机上运行远程SSH服务器上的X窗口系统程序，那么登录到远程计算机上，创建一个名为“~/.ssh/environment”的文件并加上这一行：

```
XAUTHORITY=/home/[remote user name]/.Xauthority
```

比如启动一个X程序（xterm）可以使用这个命令：

```
ssh -f -X -l [remote user name] [remote machine] xterm
```

这将在远程运行xterm程序。其他的X程序也是用相同的方法。

#### 3．为 Linuxconf 加上“加密通道”

Linuxconf是Linux的配置工具，支持远程管理。使用linuxconf的命令为：

```
remadmin --exec [link_command] linuxconf --guiproto
```

如果用户想在两台计算机之间用加密的方式传送信息，那么最好使用ssh，命令是：

```
remadmin --exec ssh -l [account] linuxconf --guiproto
```

这是一种非常有效的采用图形界面管理计算机的方式。这种方法需要在客户端安装linuxconf。其他的方法还有直接登录到服务器上用“X11Forwarding”或字符界面运行linuxconf。

## 9.5 应用三：使用 OpenSSL 进行应用层加密

通常的连接方式中，通信是以非加密的形式在网络上传播的，这样就有可能被非法窃听到，尤其是用于认证的口令信息。为了避免这个安全漏洞，就必须对传输过程进行加密。对HTTP传输进行加密的协议为HTTPS，它是通过SSL进行HTTP传输的协议，不但通过公用密钥的算法进行加密保证传输的安全性，而且还可以通过获得认证证书CA，保证客户连接的服务器没有被假冒。

SSL协议使用通信双方的客户证书以及CA根证书，允许客户/服务器应用以一种不能被偷听的方式通信，在通信双方间建立起了一条安全的、可信任的通信通道。它具备以下基本特征：信息保密性、信息完整性、相互鉴定。该协议主要使用Hash编码、加密技术。

具体使用OpenSSL机制对通信进行加密的方法已经在第 7 章进行了详细讲述，这里不再赘述。

## 9.6 数据防泄露技术原理及其应用

数据泄露防护（Data Leakage Prevention，DLP），又称为“数据丢失防御”（Data Loss Prevention，DLP），有时也称为“信息泄露防御”（Information Leakage Prevention，ILP）。数据泄露防护是通过一定的技术手段，防止企业的指定数据或信息资产以违反安全策略规定的形式流出企业。

通常认为，DLP在实施之前分六个步骤进行准备。

（1）企业先将数据进行分类，同时对“涉密数据”进行定义，然后确定哪些数据需要保护。

（2）确定涉密数据在企业系统中的存放位置，企业应明确有多少数据存放在员工的计算机、公司的服务器或数据库等。

（3）清楚掌握数据的位置，便能为数据的流出、流入提供实时的监控及保护，包括经电子邮件、http、即时消息等途径发放的资料。

（4）数据泄露很多都是人为所致，因此企业必须制定员工传送机密数据的权限。

（5）企业亦需监控数据被送达的地方是否安全，如商业伙伴或网上电子邮件等。

（6）企业必须注意员工运用什么途径传送档案，这些途径包括所有移动储存硬盘和点对点传送等。

完整执行上述六个步骤，基本可以对企业的机密资料做出全面梳理，为实施DLP做好基础性工作。把涉密信息与非涉密信息区分开来，采用等级管理办法，针对不同的信息分别进行防护。

在商业软件领域，亿赛通、websense等公司都有自己的产品。而在开源软件领域，这方面的几乎没有，在本文推荐一下由Goolge主导的OpenDLP。

OpenDLP是一个免费/开放源代码、基于代理/无代理、集中式、大规模分派的数据防泄露工具。通过一个集中式的Web应用程序和使用恰当Windows、UNIX、MySQL、MSSQL的凭据，OpenDLP能够识别出数以千计的Windows操作系统、UNIX操作系统、MySQL数据库和MSSQL数据库的敏感数据泄露问题。通过测试，目前该产品还不是非常成熟，仅仅具备定位和发现被管理机器上敏感数据的泄露功能，但是不能进行控制和阻断，因此可以理解为一个雏形。

在使用过程中，可以从网站http://code.google.com/p/opendlp/downloads/list下载已经准备好的虚拟机，然后按照README文件的指导，进行如下几个步骤的操作。

（1）从http://www.virtualbox.org下载并安装virtualbox虚拟机软件。

（2）将下载的OpenDLP虚拟机导入到virtualbox中。

（3）从Windows系统拷贝sc.exe文件到虚拟机中的/var/www/OpenDLP/bin目录。

（4）将client.p12 文件导入到你的IE浏览器中，作为数字证书。

这样就可以体验效果了。具体如下。

① 打开Web控制端，进行扫描设定，如图 9-21 所示。

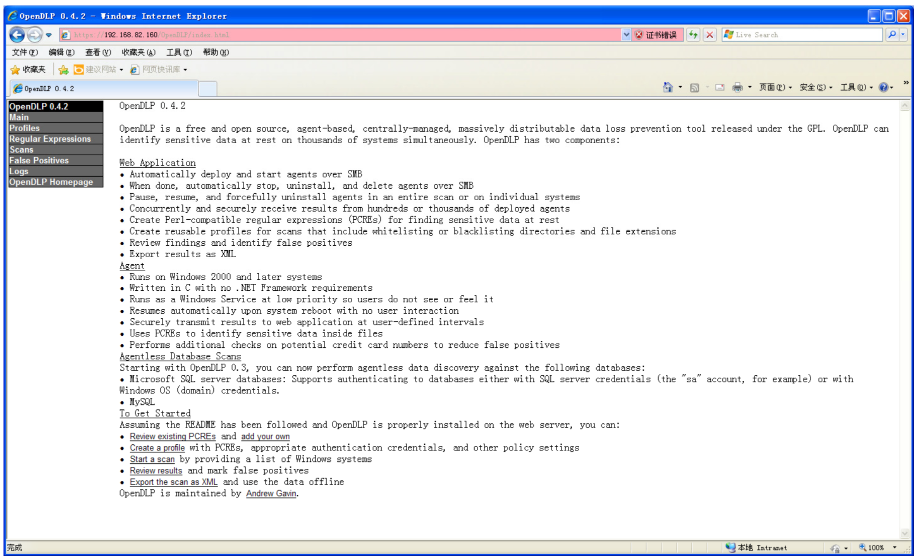



图 9-21　扫描设定

② 进行扫描设定，配置profile，然后启动扫描，如图 9-22 所示。

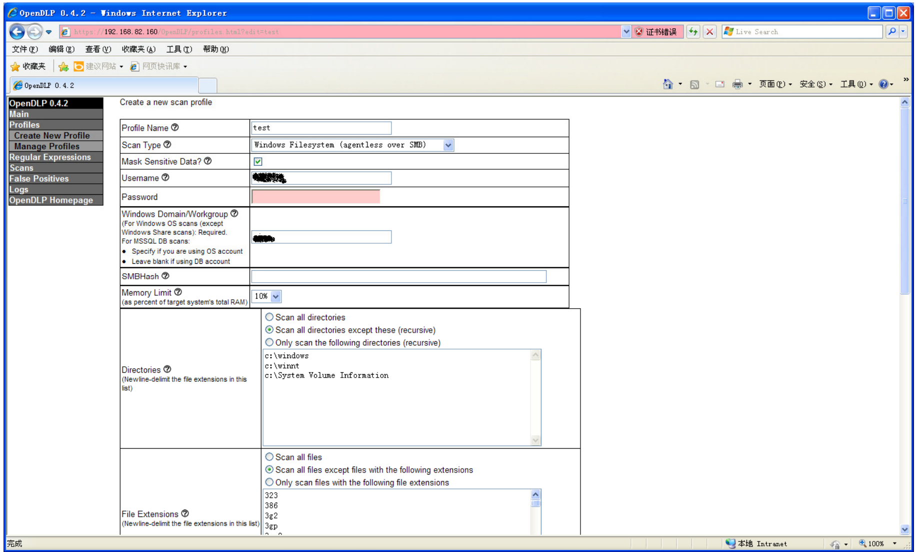
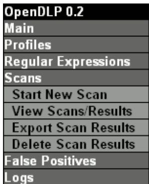



图9-22　启动扫描

③ 查看扫描结果，如图 9-23、图 9-24 所示。




View Results

Select a system to view its results for scan "test_4":

| | Network name | IP address | Status | Step | Files done | Total files | Bytes done | Total bytes | Updated | Findings | Pause | Resume | Uninstall |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ⦿ | WINDOWS | 192.168.1.121 | running | 2: Scanning | 19614 | 29984 | 931630726 | 1764587173 | 00:01:40 ago | 74 | Pause | N/A | Uninstall |
| | | | | | | | | | 52.80% done, approx 00:09:58 remaining | | | | |
| ○ | BOHNAM | 192.168.1.119 | running | 2: Scanning | 991 | 11351 | 501778250 | 1949811839 | 00:02:58 ago | 104 | Pause | N/A | Uninstall |
| | | | | | | | | | 25.73% done, approx 00:15:35 remaining | | | | |

View Results

图9-23　扫描结果（1）

## View Results

Results for database server 192.168.1.127:

| Profile | mssql_os_auth |
|---|---|
| Status | finished |
| Step | 3: Done |
| Databases Done | 2 |
| Databases Total | 2 |
| Tables Done | 2 |
| Tables Total | 2 |
| Columns Done | 7 |
| Columns Total | 7 |
| Progress | |
| Percentage | 100% |
| Completion Time | |
| Total Findings | 10 |
| False Positives | 0 |
| Valid Findings | 10 |
| Updated | 00:00:08 ago |
| Pause | N/A |
| Resume | N/A |
| Stop and Uninstall | N/A |

| # | Regex | Pattern | Database | Table | Column | Row | False? |
|---|---|---|---|---|---|---|---|
| 1 | Social_Security_Number_dashes | 423-12-4591 | users | users | ssn | 1 | ☐ |
| 2 | Social_Security_Number_dashes | 523-23-5521 | users | users | ssn | 2 | ☐ |
| 3 | Social_Security_Number_dashes | 242-34-1003 | users | users | ssn | 3 | ☐ |
| 4 | Social_Security_Number_dashes | 875-23-6547 | users | users | ssn | 4 | ☐ |
| 5 | Social_Security_Number_dashes | 521-12-1234 | users | users | ssn | 5 | ☐ |
| 6 | Social_Security_Number_dashes | 654-14-3456 | users | users | ssn | 6 | ☐ |
| 7 | Social_Security_Number_dashes | 456-34-1235 | users | users | ssn | 7 | ☐ |
| 8 | Social_Security_Number_dashes | 765-23-0834 | users | users | ssn | 8 | ☐ |

图9-23　扫描结果（2）

# 第 10 章

# 通道保障：企业移动通信数据防护

## 本章导读

随着网络通信技术的发展和网络应用的不断涌现，500 强企业不可避免地存在越来越多的用户数据和企业信息在互联网进行传送。随之而来的是越来越多的黑客和网络威胁，他们对这些机密、敏感的数据采用各种手段进行窃取、篡改和破坏，从而达到其不可告人的目的。因此，通信数据的安全性受到前所未有的挑战。鉴于这个目的，保证数据传输安全的 VPN 技术应运而生。

因此，本章将详细介绍 VPN 技术的基本原理和分类，并深入分析和探讨如何使用开源的 VPN 技术来保障 500 强企业的移动数据通信安全。

# 10.1 VPN 使用需求分析

## 10.1.1 VPN 简介

VPN（Virtual Private Network），虚拟专用网，指在公共网络（如Internet）上构建临时的、安全的逻辑网络的技术，如图 10-1 所示。

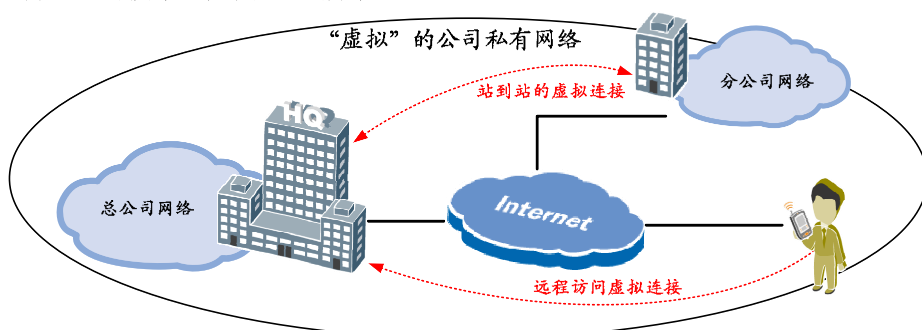



图10-1　VPN示意图

### 1. 节省费用

机构遍布各地的公司，尤其跨越国家的公司，可以通过VPN接入总部网络。VPN利用了现有的公共网络资源，从而节省了公司租用运营商跨省、跨海专线的费用。分支机构网络通过VPN接入总部后，就好比与总部网络接入同一个局域网，可访问总部网络能访问的各项资源，如图 10-2 所示。

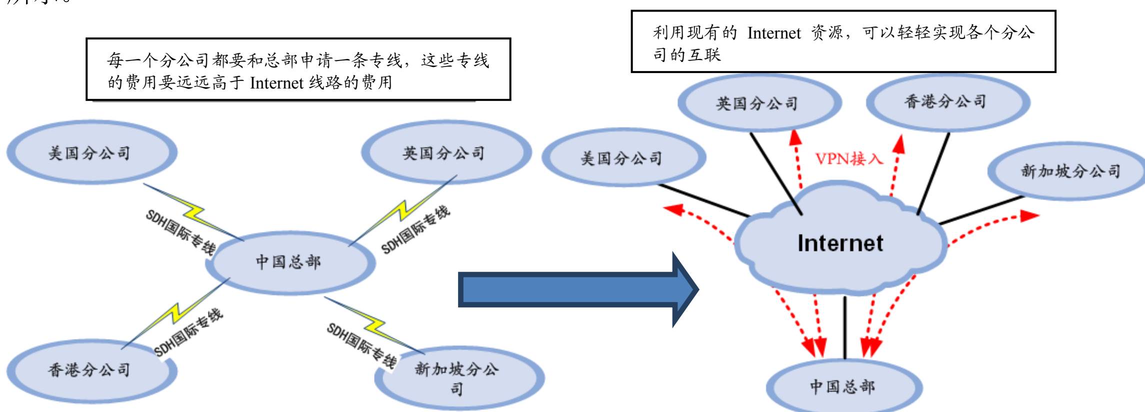



图10-2　VPN连接示意图

### 2. 安全保密

为保证数据不在网络（尤其是公共网络）上被窃取，通过VPN技术实现数据加密传送。需要澄清的一点是，VPN技术有多种，并非所有的VPN标准均具有高安全性，如图 10-3 所示。

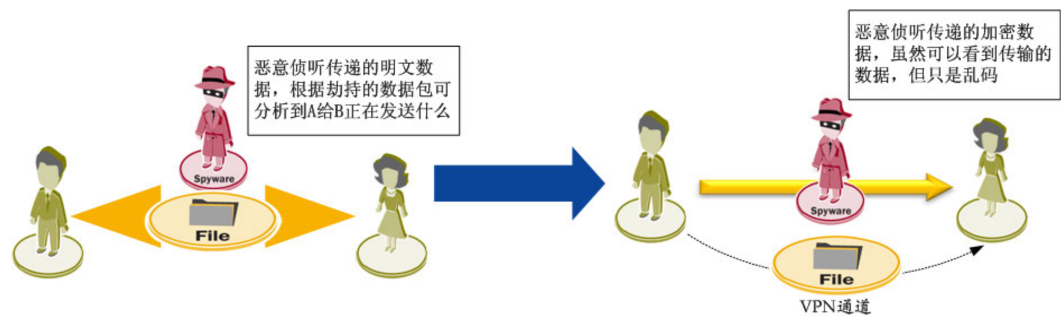



图10-3　VPN安全示意图

## 10.1.2　VPN 安全技术分析

目前VPN主要采用四项技术来保证安全，这四项技术分别是隧道技术（Tunneling）、加/解密技术（Encryption & Decryption）、密钥管理技术（Key Management）、使用者与设备身份认证技术（Authentication）。

### 1．隧道技术

隧道技术是VPN的基本技术，类似于点对点连接技术，它在公用网络上建立一条安全的数据通道（隧道），让数据包通过这条隧道进行传输。

隧道是由隧道协议形成的，分为第二、三层隧道协议，对应TCP/IP协议栈中的数据链路层和网络层。第二层隧道协议是先把各种网络协议封装成为第二层帧，再把整个帧装入隧道协议中。这种双层封装方法形成的数据包靠第二层协议进行传输，如图 10-4 所示。

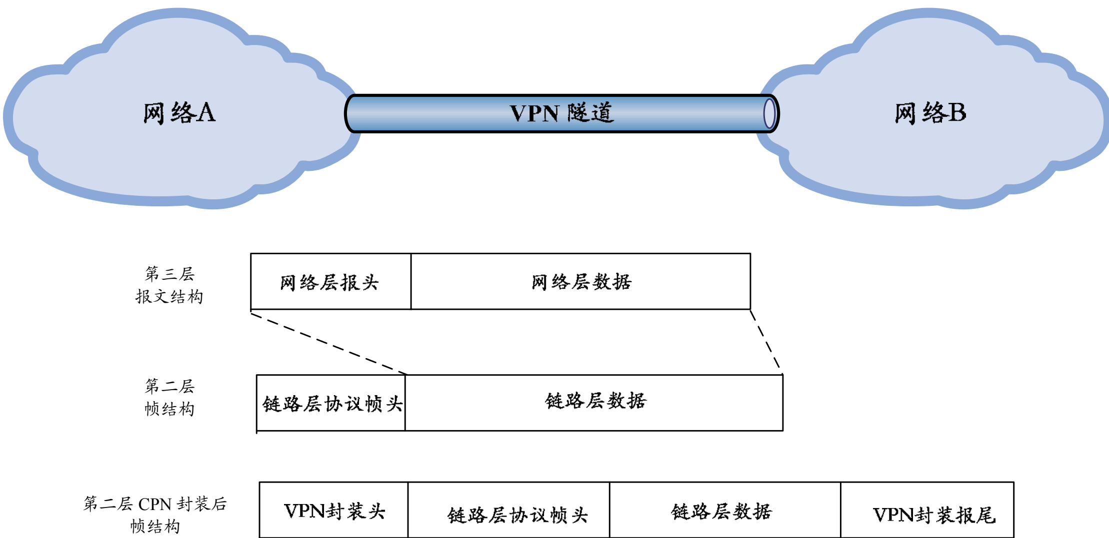



图 10-4　VPN 隧道示意

第三层隧道协议是把各种网络协议直接装入隧道协议中，形成的数据包依靠第三层协议进行传输。第三层隧道协议有IPSec等，如图 10-5 所示。

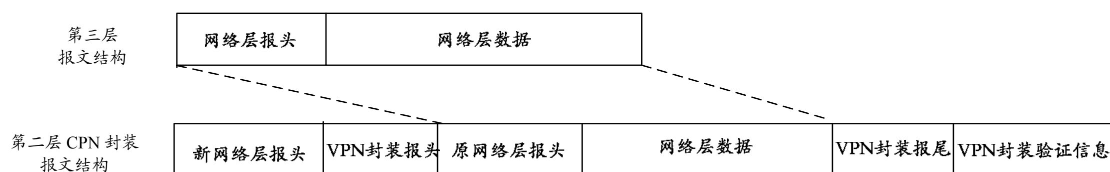



图 10-5 VPN 封装报文结构

### 2．加/解密技术

不同的VPN协议采用的加密方式不同，但所有的加密算法可归结为两大类：对称式加密算法和非对称式加密算法。

对称式加密就是加密和解密使用同一个密钥，通常称之为“Session Key”，这种加密技术目前被广泛采用。对称式加密原理简单，加/解密速度快；但密钥的分发过程（如在网络上传输）本身就存在泄露的可能性，带来安全性问题。

非对称式加密就是加密和解密所使用的不是同一个密钥，通常有两个密钥，称为“公钥”和“私钥”，它们两个必须配对使用。这里的“公钥”是指可以对外公布的，“私钥”则不能，只能由持有人一个人知道。通信双方使用对方的公钥对发送的传输内容进行加密，而利用自己的私钥对接收的传输内容进行解密。由于解密的私钥是不传输的，这样就很好地避免了密钥分发带来的安全性问题。

如MPPE（Microsoft Point-to-Point Encryption，微软专有技术，一种对数据链路层的加密技术）加密使用 40～128 位密钥，使用对称加密算法；IPSec采用 64 位或者 3 组 64 位密钥，使用对称加密算法；SSL采用 64～128 位加密密钥，使用非对称加密算法。

一般来说，密钥的位数越长，越难以被破解。因此，综合分析采用 128 位的SSL加密技术或采用 3 组 64 位加密密钥的IPSec是最为安全的。

### 3．密钥管理技术

密钥管理技术的主要任务是如何在公用数据网上安全地传递密钥而不被窃取。现行密钥管理技术分为SKIP与ISAKMP两种。

SKIP主要是利用Diffie-Hellman的演算法则，在网络上传输密钥；以两人Alice和Bob通信为例，Diffie-Hellman密钥交换协议如下。

首先，Alice和Bob双方约定两个大整数：素数p及其原根g，其中 1<g<p，这两个整数无须保密，然后执行下面的过程。

（1）Alice随机选择一个大整数x（保密），并计算X=gx mod p。

（2）Bob随机选择一个大整数y（保密），并计算Y=gy mod p。

（3）Alice把X发送给Bob，Bob把Y发送给Alice。

（4）Alice计算出K=Yx mod p。

（5）Bob计算出K=Xy mod p。

K即是共享的密钥。双方计算出相同的K，这是利用了数学的性质。

假设监听者Eve在网络上只能监听到X和Y，但无法通过X，Y计算出x和y，因此，Eve无法计算出K。

ISAKMP使用了前述的非对称加密算法，双方都有两把密钥，分别用于公用、私用。比如有

两个用户Alice和Bob，Alice想把一段明文通过双钥加密的技术发送给Bob，Bob有一对公钥和私钥，那么加密和解密的过程如下。

（1）Bob将他的公开密钥传送给Alice。

（2）Alice用Bob的公开密钥加密她的消息，然后传送给Bob。

（3）Bob用他的私人密钥解密Alice的消息。

#### 4．使用者与设备身份认证技术

使用者与设备身份认证技术最常用的是使用者名称、密码或令牌认证等方式。通过VPN接入设备与Active Directory服务器、LDAP服务器或者动态密码服务器集成，提高认证的准确性与安全性，如图 10-6 所示。

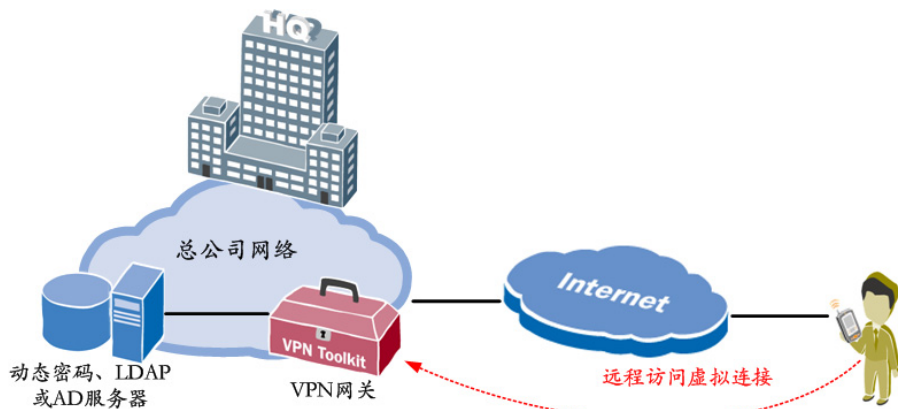



图10-6　VPN与其他基础设施的集成

## 10.2　Linux 提供的 VPN 类型

### 10.2.1　IPSec VPN

IPSec（IP Security）是为实现VPN 功能而最普遍使用的协议。通过相应的隧道技术，可实现VPN。IPSec有两种模式：隧道模式和传输模式。

IPSec 不是一个单独的协议，它给出了应用于IP层上网络数据安全的一整套体系结构。该体系结构包括认证头协议（Authentication Header，简称为AH）、封装安全负载协议（Encapsulating Security Payload，简称为ESP）、密钥管理协议（Internet Key Exchange，简称为IKE）和用于网络认证及加密的一些算法等。

IPSec 规定了如何在对等体之间选择安全协议、确定安全算法和密钥交换，向上提供了访问控制、数据源认证、数据加密等网络安全服务。

- 认证头协议（AH）。IPSec 体系结构中的一种主要协议，它为 IP 数据包提供无连接完整性与数据源认证，并提供保护以避免重播情况。AH 尽可能为 IP 头和上层协议数据提供足够多的认证。
- IPSec 封装安全负载（ESP）。IPSec 体系结构中的一种主要协议。ESP 加密需要保护的数据并且在 IPSec ESP 的数据部分进行数据的完整性校验，以此来保证机密性和完整

性。ESP 提供了与 AH 相同的安全服务并提供了一种保密性（加密）服务，ESP 与 AH 各自提供的认证的根本区别在于它们的覆盖范围。

- 密钥管理协议（IKE）。一种混合型协议，由 Internet 安全联盟（SA）和密钥管理协议（ISAKMP）这两种密钥交换协议组成。IKE 用于协商 AH 和 ESP 所使用的密码算法，并将算法所需的必备密钥放到恰当位置。

该技术的优点是定义了一套用于认证、保护私有性和完整性的标准协议。IPSec支持一系列加密算法，如DES、三重DES、IDEA。它检查传输的数据包的完整性，以确保数据没有被修改。IPSec用来在多个防火墙和服务器之间提供安全性。IPSec可确保运行在TCP/IP协议上的VPNs之间的互操作性。该技术的缺点是IPSec在客户机/服务器模式下实现有一些问题，在实际应用中，需要公钥来完成。IPSec需要已知范围的IP地址或固定范围的IP地址，因此在动态分配IP地址时不太适合于IPSec。除了TCP/IP协议外，IPSec不支持其他协议。另外配置比较复杂。

在具体的使用中，Linux实现使用IPSec的软件是：Free S/WAN http://www.freeswan.org/。FreeS/WAN不支持NAT（Network Address Translation，网络地址翻译）和IP地址伪装用于加密通道通信。

### 10.2.2　PPP Over SSH

SSH是一种基于安全会话目的的应用程序。SSH支持身份认证和数据加密，对所有传输的数据进行加密处理。同时，可以对传输数据进行压缩处理，以加快数据传输速度。SSH既可以代替Telnet作为安全的远程登录方式，又可以为FTP、POP等提供一个安全的“隧道”。OpenSSH是SSH的替代软件包，是免费的。用PPP端口在SSH上运行技术实现VPN的方法，其优点是安装配置简单；缺点是运行时系统开销比较大。PPP Over SSH的具体应用软件有SSHVNC（http://3sp.com/products/sshtools/sshvnc/sshvnc.php）。

### 10.2.3　CIPE：Crypto IP Encapsulation

CIPE（加密IP封装）是主要为Linux而开发的VPN实现。CIPE使用加密的IP分组，这些分组被封装或“包围”在数据报（UDP）分组中。CIPE分组被给以目标头信息，并使用默认的CIPE加密机制来加密。CIPE使用标准的Blowfish或IDEA加密算法来支持加密。根据你所在国家的加密出口法规而定，你可以使用默认方法（Blowfish）来加密你的专用网上的所有CIPE交通。CIPE配置可以通过文本文件、图形化的网络管理工具来完成。用CIPE技术实现VPN的方法，其优点是安装配置简单，运行时系统开销比较小；缺点是CIPE不是一种标准的VPN协议，不能支持所有平台。CIPE的网址为：http://sites.inka.de/~。

### 10.2.4　SSL VPN

IPSec VPN和SSLVPN是两种不同的VPN架构，IPSec VPN是工作在网络层的，提供所有在网络层上的数据保护和透明的安全通信，而SSL VPN是工作在应用层（基于HTTP协议）和TCP层之间的，从整体的安全等级来看，两者都能够提供安全的远程接入。但是，IPSec VPN技术是被设计用于连接和保护在信任网络中的数据流，因此更适合为不同的网络提供通信安全保障，而SSL VPN则更适合应用于远程分散移动用户的安全接入。一般来说，SSL VPN相对于IPSec VPN部署和实施成本低。在设计上，IPSec VPN是一种基础设施性质的安全技术。这类VPN的真正价

值在于，它们尽量提高IP环境的安全性。可问题在于，部署IPSec需要对基础设施进行重大改造，以便远程访问。好处就摆在那里，但管理成本很高。IPSec安全协议方案需要大量的IT技术支持，包括在运行和长期维护两个方面。在大的企业通常有几个专门的员工为通过IPSec安全协议进行的VPN远程访问提供服务。IPSec VPN最大的难点在于客户端需要安装复杂的软件，而且当用户的VPN策略稍微有所改变时，VPN的管理难度将呈几何级数增长。SSL VPN则正好相反，客户端不需要安装任何软件或硬件，使用标准的浏览器，就可通过简单的SSL安全加密协议，安全地访问网络中的信息。SSL VPN避开了部署及管理必要客户软件的复杂性和人力需求；SSL在Web的易用性和安全性方面架起了一座桥梁，目前对SSL VPN公认的三大好处如下。

- 简单。它不需要配置，可以立即安装、立即生效。
- 客户端不需要麻烦的安装，直接利用浏览器中内嵌的SSL协议就行。
- 兼容性好。传统的IPSec VPN对客户端采用的操作系统版本具有很高的要求，不同的终端操作系统需要不同的客户端软件，而SSL VPN则完全没有这样的麻烦。

因此，SSL VPN强调的优势其实主要集中在VPN客户端的部署和管理上，我们知道SSL VPN一再强调无须安装客户端，主要是由于浏览器内嵌了SSL协议，也就是说是基于B/S结构的业务时，可以直接使用浏览器完成SSL的VPN建立。

OpenVPN 是一个基于OpenSSL库的应用层VPN实现。详细信息可以参考http://www.openvpn.net。OpenVPN的优点是支持多种常用应用系统。目前版本支持Linux。Windows 2000/XP and higher、OpenBSD、FreeBSD、NetBSD、Mac OS X、Solaris。支持多种客户端连接模式。可以通过GUI便捷地操纵OpenVPN工作在OSI layer 2 或 3 使用标准的SSL/TLS协议。可以通过certificates或smart cards认证。加密强度较高，不易在传输通路上被人劫持而破解信息资源。OpenVPN的缺点是使用SSL应用层加密，传输效率要低于IPSec传输的VPN软件。

### 10.2.5 PPPTD

点对点隧道协议（PPTP）是一种支持多协议虚拟专用网络的网络技术。PPTP可以用于在IP网络上建立PPP会话隧道。在这种配置下，PPTP隧道和PPP会话运行在两个相同的机器上，呼叫方充当PNS。PPTP使用客户机－服务器结构来分离当前网络访问服务器具备的一些功能并支持虚拟专用网络。PPTP作为一个呼叫控制和管理协议，它允许服务器控制来自PSTN或ISDN的拨入电路交换呼叫访问并初始化外部电路交换连接。PPTP只能通过PAC和PNS来实施，其他系统没有必要知道PPTP。拨号网络可与PAC相连接而无须知道PPTP。标准的PPP客户机软件可继续在隧道PPP连接上操作。PPTP使用GRE扩展版本来传输用户PPP包。这些增强允许为在PAC和PNS之间传输用户数据的隧道提供低层拥塞控制和流控制。这种机制允许高效使用隧道可用带宽并且避免了不必要的重发和缓冲区溢出。PPTP没有规定特定的算法用于低层控制，但它确实定义了一些通信参数来支持这样的算法工作。PPTP相对其他远程“拨入”型VPN的不凡之处在于微软Windows（95/98/Me/NT/2000/XP/Vista）拥有一个内置的PPTP客户端，这意味着管理员不必涉及任何额外的客户端软件以及那些通常伴随出现的问题。Linux PPTP服务器实现的软件是：Poptop（http://www.poptop.org/），开源PPTP服务器产品Poptop特性为：

- 微软兼容的认证和加密（MSCHAPv2，MPPE40～128位RC4加密）。
- 支持多个客户端连接。
- 使用RADIUS插件无缝集成到一个微软网络环境中。

- 和 Windows 95/98/Me/NT/2000/XP PPTP 客户端共同工作。
- 和 Linux PPTP 客户端共同工作。
- Poptop 在 GNU 通用公共许可下是，并仍将是完全免费。

# 10.3 使用 OpenVPN 构建 SSL VPN

## 10.3.1 OpenVPN 简介

OpenVPN是一个强大、高度可配置、基于SSL的VPN（Virtual Private Network）Open Source软件。它具有多种验证方式以及许多强大的功能。OpenVPN工作在OSI模型的第 2 层或第 3 层，使用SSL/TLS协议进行网络传输。支持多种客户认证方法，如证书、smart cards，加上用户名密码的证书认证等。除此以外，还有强大的ACL功能限制客户的信息交换。

OpenVPN可以运行在多种操作系统中，包括：Linux、Windows 2000/XP and higher、OpenBSD、FreeBSD、NetBSD、Mac OS X、Solaris。通过使用OpenVPN，可以实现：

- 使用特定 UDN 或 TCP 端口实现两台主机之间的 VPN 连接。
- 实现 C/S 结构，实现多台 Client 通过 Server 服务器互连互通。
- 通过 TLS/SSL 加密保证数据传输的安全。
- 通过数据的压缩，提高数据传输的速度。

## 10.3.2 安装 OpenVPN

企业级Linux版本的安装光盘中自带了OpenVPN的安装程序，在系统安装的时候，用户可以选择进行安装。如果系统安装时没有安装，用户也可以随时使用安装盘进行安装。为了确认系统是否已经安装该软件，可以使用如下命令进行查看：

```
#rpm -qa | grep openvpn
```

## 10.3.3 制作证书

### 1．制作证书前的准备

复制OpenVPN证书工具包，在安装完OpenVPN后，系统会在/etc下建一个OpenVPN的目录，这样我们可以把OpenVPN证书工具包拷贝到/etc/openvpn目录下，需要注意以下几个主要的存放位置。

- 证书工具包默认位置：/usr/share/openvpn/easy-rsa。
- 准备配置证书位置：/etc/openvpn/。
- 证书生成位置：/etc/openvpn/easy-rsa/2.0/keys。

使用命令如下：

```
# cp -r /usr/share/openvpn/easy-rsa /etc/openvpn/
# mkdir /etc/openvpn/easy-rsa/2.0/keys
```

### 2．修改 vars 变量初始化配置文件

编辑easy-rsa/2.0/vars文件，需要进行以下几个操作。

- 注释掉 export CA_EXPIRE=3650，在前面加个“#”号即可。
- 注释掉 export KEY_EXPIRE=3650，在前面加个“#”号即可。
- 修改证书默认值，如图 10-7 中白色区域所示。

```
# Increase this to 2048 if you
# are paranoid.  This will slow
# down TLS negotiation performance
# as well as the one-time DH parms
# generation process.
export KEY_SIZE=1024

# In how many days should the root CA key expire?
export CA_EXPIRE=3650

# In how many days should certificates expire?
export KEY_EXPIRE=3650

# These are the default values for fields
# which will be placed in the certificate.
# Don't leave any of these fields blank.
export KEY_COUNTRY="CN"
export KEY_PROVINCE="BJ"
export KEY_CITY="Beijing"
export KEY_ORG="tsinghua publishing house"
export KEY_EMAIL="publishing_house@tsinghua.com"
```

图10-7　修改vars文件中的相应区域

### 3．初始化证书库

初始化证书库主要包括初始化变量库和清空变量库两个步骤，如下命令所示：

```
//初始化变量库
#./vars
//清空证书库
#./clean-all
```

### 4．证书验证机制

执行命令，如图 10-8 所示。在执行过程中，输入对应的信息，完成后在keys目录下会生成ca.crt和ca.key两个文件。

```
[root@localhost 2.0]# ./build-ca
Generating a 1024 bit RSA private key
.......................++++++
..........++++++
writing new private key to 'ca.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [CN]:
State or Province Name (full name) [BJ]:
Locality Name (eg, city) [Beijing]:
Organization Name (eg, company) [tsinghua publishing house]:
Organizational Unit Name (eg, section) []:vpn
Common Name (eg, your name or your server's hostname) [tsinghua publishing hou
se CA]:vpn_server
Email Address [publishing_house@tsinghua.com]:
```

图10-8　完成证书验证机制

### 5．生成服务器证书

执行./build-key-server server命令，执行完成后在keys目录下，会生成server.crt、server.csr和server.key三个文件，如图 10-9 所示。

```
.....................................++++++
....++++++
writing new private key to 'server.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [CN]:
State or Province Name (full name) [BJ]:
Locality Name (eg, city) [Beijing]:
Organization Name (eg, company) [tsinghua publishing house]:
Organizational Unit Name (eg, section) []:vpn
Common Name (eg, your name or your server's hostname) [server]:vpn_server
Email Address [publishing_house@tsinghua.com]:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:duanjia
An optional company name []:
Using configuration from /etc/openvpn/easy-rsa/2.0/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
```

图10-9　生成服务器证书

### 6．生成客户端证书

在OpenVPN中，这种配置方法是每一个登录的VPN客户端需要有一个证书，每个证书在同一时刻只能供一个客户端连接（如果有两个机器安装相同证书，同时拨服务器，都能拨上，但是只有第一个拨上的才能连通网络）。所以需要建立许多份证书。执行./build-key vpn_client1 命令，执行完成后在keys目录下，会生成vpn_client1.crt、vpn_client1.csr和vpn_client1.key三个文件，如图 10-10 所示。

```
[root@localhost 2.0]# ./build-key vpn_client1
Generating a 1024 bit RSA private key
.................++++++
...........++++++
writing new private key to 'vpn_client1.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [CN]:
State or Province Name (full name) [BJ]:
Locality Name (eg, city) [Beijing]:
Organization Name (eg, company) [tsinghua publishing house]:
Organizational Unit Name (eg, section) []:vpn
Common Name (eg, your name or your server's hostname) [vpn_client1]:
Email Address [publishing_house@tsinghua.com]:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:vpn_client1
An optional company name []:
Using configuration from /etc/openvpn/easy-rsa/2.0/openssl.cnf
```

图10-10　为客户端vpn_client1生成客户端证书

一个服务器端可以配置多个客户端证书，但是每个客户端都不能一样，主要体现在Common Name这里，每一个客户端的Common Name都必须不一样，用户可以再继续建出多个客户端证书，如下所示：

```
#./build-key vpn_client2
#./build-key vpn_client3
```

### 7．编译证书

采用./build-dh命令将刚刚生成的证书和密钥等进行一次加密算法，然后证书和密钥都准备好了，此时在keys目录下，会生成dh1024.pem文件，如图 10-11 所示。

```
[root@localhost 2.0]# ./build-dh
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
...+...............+...........+...+.............................................
..................................................+..............................
.................................................................................
.............................+..........................................+........
................................+............+...........................+.......
...................
```

图10-11　编译证书

## 10.3.4　配置服务端

如果上一小节中所述的七步都运行正常的情况下，用户就可以开始配置服务端了；如果中间过程出现问题，建议清空证书库，重新操作，执行./clean-all来清空库即可。

配置服务端需要执行以下几个步骤。

（1）把几个证书、钥匙和验证机制都拷贝到/etc/openvpn目录下，命令如下：

```
# cp /etc/openvpn/easy-rsa/2.0/keys/ca.crt /etc/openvpn/
# cp /etc/openvpn/easy-rsa/2.0/keys/dh1024.pem /etc/openvpn/
# cp /etc/openvpn/easy-rsa/2.0/keys/server.crt /etc/openvpn/
# cp /etc/openvpn/easy-rsa/2.0/keys/server.key /etc/openvpn/
```

（2）把服务器的配置文件拷贝到/etc/openvpn目录下：

```
# cp /usr/share/doc/openvpn-2.1/sample-config-files/server.conf /etc/openvpn/
```

（3）编辑/etc/openvpn/server.conf配置文件，去掉下述选项前面的分号“;”，以使其生效：

```
;user nobody
;group nobody
```

（4）启动OpenVPN，查看一下端口 1194 是否已开放，如果 1194 已开放，那就表明OpenVPN服务端已经配置成功了：

```
#/etc/init.d/openvpn start
```

（5）设置开机自启动OpenVPN，使用如下命令，找到OpenVPN服务，（见图 10-12），并选中服务确认即可，下次系统启动的时候会自动加载OpenVPN服务：

```
#ntsysv
```

（6）设置防火墙，使得VPN流量能够正常通过。在系统管理中，打开防火墙配置，开启OpenVPN和SSH（前面打勾即可），如图 10-13 所示。

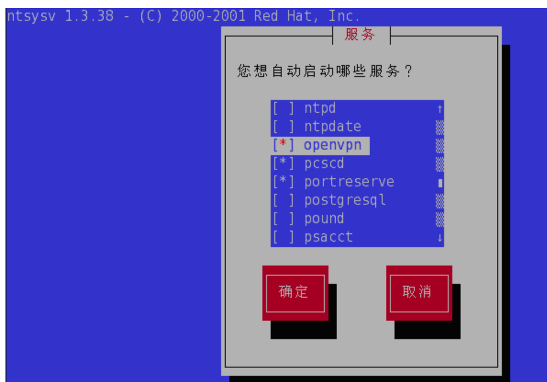



图10-12　设置OpenVPN开机启动




图10-13　设置防火墙

## 10.3.5　配置客户端

配置客户端vpn_client1 需要使用源代码包目录sample-config-files里的client.conf 修改即可，如下片段所示：

```
client
dev tun
proto tcp
remote IP_address port
resolv-retry infinite
nobind
user nobody
group nobody
persist-key
persist-tun
ca ca.crt
cert vpn_client1.crt。
key vpn_client1.key
#comp-lzo
```

其中，主要考虑以下几个参数的设定。

- proto tcp 或 proto udp：和 server.conf 一致即可。

- remote IP_address port：指定 VPN Server 的 IP 地址和端口。
- cert vpn_client1.crt 和 key vpn_client1.key：这两行添入 vpn_client1 生成的密钥文件。

通过以上配置后，就可以使用OpenVPN进行安全通信了。

### 10.3.6 一个具体的配置实例

假设要搭建一个VPN服务，VPN服务端有两个IP地址，其中eth0（172.16.6.79）提供VPN服务，由外部 172.31.16.0/24 通过 1194 端口拨入，然后通过eth1（192.1610.253.79）访问192.1610.253.0/24 去维护服务器。

具体的配置如下：

#### 1．服务器端配置

VPN Server 的目录/etc/openvpn 下有文件ca.crt、ca.key、dh1024.pem、server.crt、server.key、server.conf 以及子目录ccd。/etc/openvpn/ccd目录下有文件client1，client2 和client3。

（1）/etc/openvpn/server.conf 内容如下：

```
;local a.b.c.d
port 1194
proto tcp
dev tun
ca ca.crt
cert server.crt
key server.key # This file should be kept secret
dh dh1024.pem
server 10.10.0.0 255.255.255.0
ifconfig-pool-persist ipp.txt
keepalive 10 120
comp-lzo
user nobody
group nobody
persist-key
persist-tun
status openvpn-status.log
verb 3
client-config-dir ccd
#使 vpn clients 能访问 VPN Server 内部网段计算机
push "route 172.16.6.0 255.255.255.0"
route 172.31.13.0 255.255.255.0
```

（2）/etc/openvpn/ccd/client1 内容如下：

```
ifconfig-push 10.10.0.5 10.10.0.6
iroute 172.31.13.0 255.255.255.0
```

### 2. 客户机端配置

vpn client1 的目录/etc/openvpn 下有文件ca.crt、ca.key client1.crt、client1.key、client1.conf。/etc/openvpn/client1.conf内容如下：

```
Client
dev tun
proto udp
remote 172.16.6.79 1194
resolv-retry infinite
nobind
user nobody
group nobody
persist-key
persist-tun
ca ca.crt
cert client1.crt
key client1.key
comp-lzo
verb 3
keepalive 10 120
```

## 10.4 使用 IPSec VPN

### 10.4.1 安装 ipsec-tools

实现IPSec要求主机上安装ipsec-tools RPM软件包，Red Hat Enterprise Linux 5 一般默认安装。执行下述命令可以看到它的版本号为ipsec-tools-0.6.5-13.el5_3.1：

```
# rpm -qa | grep ipsec
ipsec-tools-0.6.5-13.el5_3.1
```

### 10.4.2 配置 IPSec VPN

下面以一个实际的例子来讲述如何配置IPSec VPN。假设网络A和网络B想通过IPSec隧道来彼此连接。网络A的网络地址在 172.1610.1.0/24 范围内，网络B使用 172.1610.2.0/24 范围。网络A的网关IP地址是 172.1610.1.1，网络B的网关地址是 172.1610.2.1。每个网络间的IPSec连接使用一个值为sharekey的预共享密钥，网络A和网络B都同意让racoon自动生成和共享每个IPSec路由器之间的验证密钥。网络A把IPSec连接命名为ipsec0，而网络B把IPSec连接命名为ipsec1。

上述配置可以在Red Hat Enterprise Linux 5 系统中使用图形用户界面进行方便地配置，如图 10-14～图 10-22 所示，主要步骤如下。

（1）进入IPSec VPN主界面具体方法选择“系统”→“管理”→“网络”命令，则弹出如

图 10-14 所示的对话框。

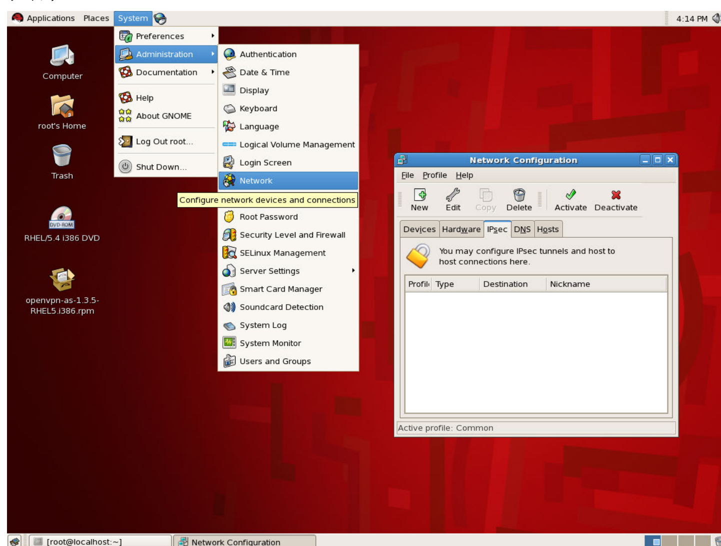



图10-14　进入IPSec VPN配置主界面

（2）选择新建一个IPSec，并输入该隧道的描述名为securecomm，如图 10-15 所示。

（3）选择IPSec VPN的连接类型，主机对主机加密或者是网络对网络的加密。后一种应用最为广泛，如图 10-16 所示。此处选择后一种方式。

（4）选择IPSec VPN的加密模式，即使用何种密钥，此处选择常用的IKA模式，如图 10-17 所示。

（5）设定IPSec VPN本地网络端的相关配置，如图 10-18 所示。

（6）设定IPSec VPN远程网络端的相关配置，如图 10-19 所示。

（7）设定IPSec共享key的名称，如图 10-20 所示。

（8）完成上述步骤后，IPSec VPN已经设置完成，将显示设置的具体信息，以供用户核对，如图 10-21 所示。

（9）IPSec VPN成功设置完成，如图 10-22 所示。




图10-15　输入IPSec VPN描述名称

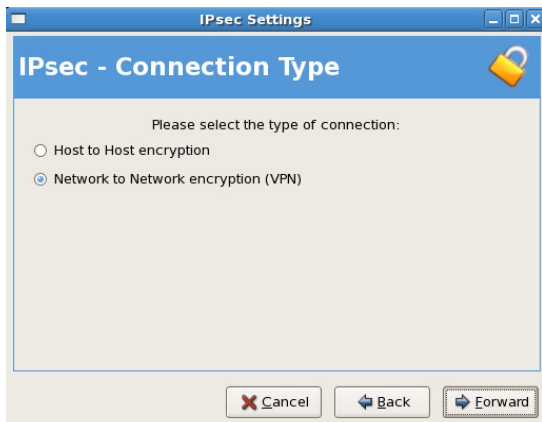



图10-16　选择连接类型

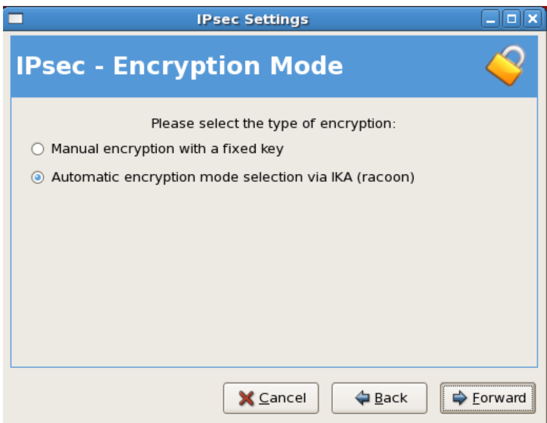



图10-17　选择采用IKA的自动加密模式

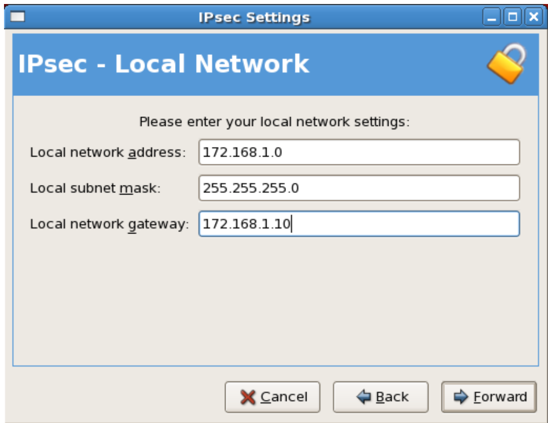



图10-18　设定本地网络配置

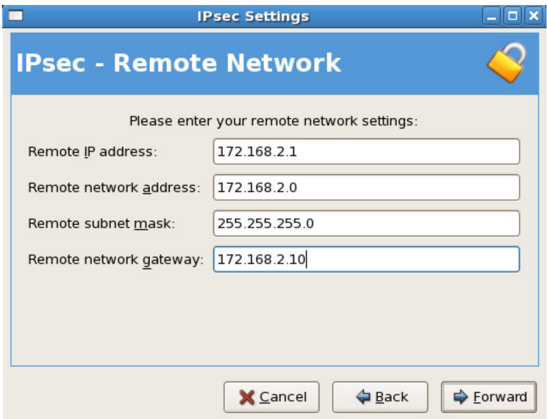



图10-19　设定远程网络配置

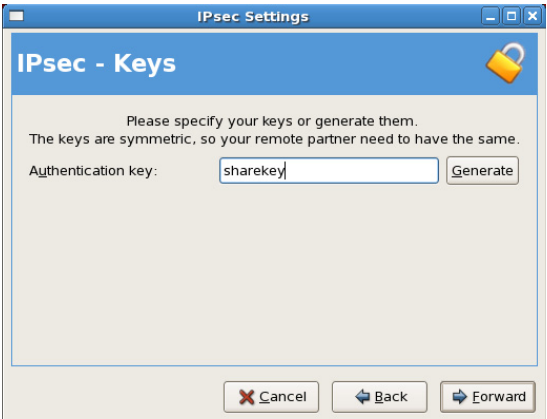



图10-20　设定IPSec共享key的名称

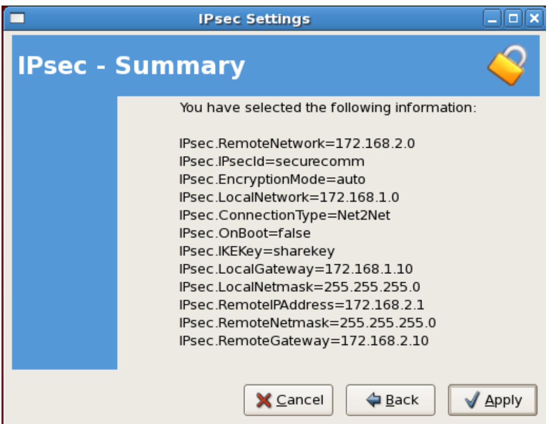



图10-21　设置完成，显示设置信息

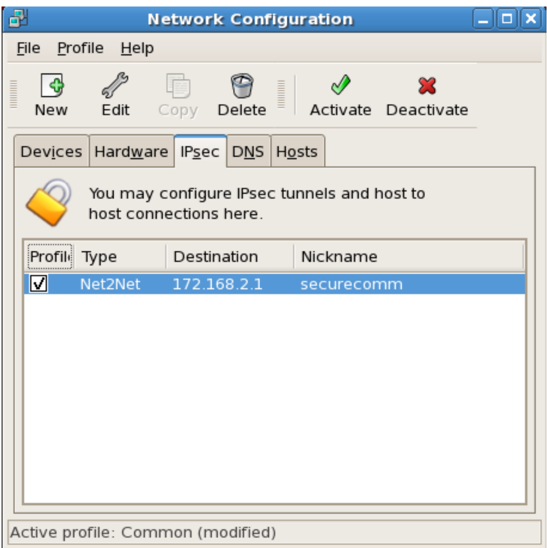



图10-22　设置好的IPSec VPN

以下是网络A的IPSec连接的ifcfg文件。在这个例子中用来识别该连接的独特名称是ipsec0，因此其结果文件被命名为/etc/sysconfig/network-scripts/ifcfg-ipsec0。

```
TYPE=IPSEC
ONBOOT=false # 引导时不激活
IKE_METHOD=PSK # 使用预共享密钥
SRCGW=172.1610.1.10 # 源网关
DSTGW=172.1610.2.10 # 目标网关
SRCNET=172.1610.1.0/24 # 源网段
DSTNET=172.1610.2.0/24 # 目标网段
DST=192.1610.1.20 # 网络 BVPN Server 的外网 IP
```

以下是预共享密钥文件（叫做 /etc/sysconfig/network-scripts/keys-ipsecX，这里的X对网络A来说是 0，对网络B来说是 1），两个工作站都使用它来彼此验证。该文件的内容应该完全一致，并且只有root用户才应该有读写权。

```
IKE_PSK=yoyotown.com
```

改变keys-ipsec0 文件的权限，只有root用户才有读写权。

```
#chmod 600 /etc/sysconfig/network-scripts/keys-ipsec0
```

要在任何时候改变验证密钥，编辑两个IPSec路由器上的keys-ipsecX文件。这两个密钥必须完全一致才能保证正确的连接性。

以下是IPSec连接的/etc/racoon/racoon.conf配置文件。

```
# Racoon IKE daemon configuration file.
# See 'man racoon.conf' for a description of the format and entries.

path include "/etc/racoon";
path pre_shared_key "/etc/racoon/psk.txt";
path certificate "/etc/racoon/certs";

sainfo anonymous
{
pfs_group 2;
lifetime time 1 hour ;
encryption_algorithm 3des, blowfish 448, rijndael ;
authentication_algorithm hmac_sha1, hmac_md5 ;
compression_algorithm deflate ;
}
include "/etc/racoon/X.X.X.X.conf" # IPSec 被激活时自动生成，X.X.X.X 为远程 VPN
Server 的外网 IP
```

以下是连接到远程网络时生成的配置文件。该文件的名称为 X.X.X.X.conf。注意，一旦IPSec隧道被激活，该文件会被自动生成。

```
remote X.X.X.X
```

```
{
exchange_mode aggressive, main;
my_identifier address;
proposal {
encryption_algorithm 3des;
hash_algorithm sha1;
authentication_method pre_shared_key;
dh_group 2 ;
}
}
```

在完成上述操作后，在启动IPSec连接前，内核中应该启用IP转发。在shell提示符下作为root用户来启用IP转发。

（1）编辑/etc/sysctl.conf，将net.ipv4.ip_forward设置为 1。

（2）执行以下命令来启用改变：

```
#sysctl -p /etc/sysctl.conf
```

（3）以root用户身份启动IPSec连接：

```
#service network restart
```

这样，IPSec连接就被激活了，网络A和网络B能够安全地进行通信了。

# 第 11 章

# 运筹帷幄：企业 Linux 服务器远程安全管理

## 本章导读

随着 500 强企业的发展，越来越多的业务、信息系统将通过 IT 技术来进行实现，大量的服务也随之而产生，结果是需要采购大量的服务器并对之进行管理。一般情况下，这些大量的服务器都是放在机房进行管理和维护。这就产生了一个问题，如何对这些服务器进行管理呢？传统的做法可能是系统管理员直接进入机房进行管理和调试，然而这种过于“亲密”的接触方式是非常不安全的，人员在机房产生的静电、污染等都会影响到服务器。并且，如果机房离管理员工位比较远的话，也会给运维带来非常大的不必要的人力开销，毕竟跑来跑去也是不小的工作量。

因此，非常合适的一种办法就是通过远程控制和管理的方法来对企业 Linux 服务器及其上面的服务进行管理，这样，管理员就能在工位轻松、高效、安全地进行调试、升级、管理等工作了。本章将详细介绍 500 强企业如何应用相关的工具来进行企业 Linux 服务器远程安全管理。

# 11.1 远程控制及管理的基本原理

谈到远程监控与管理，我们需要了解与之紧密相关的远程控制技术。远程控制技术始于 DOS 时代，只不过当时由于技术上没有什么大的变化，网络不发达，市场没有更高的要求，所以远程控制技术没有引起更多人的注意。但是，随着网络的高速发展，电脑的管理及技术支持的需要，远程操作及控制技术越来越引起人们的关注。远程控制一般支持下面的这些网络方式：LAN、WAN、拨号方式、互联网方式。此外，有的远程控制软件还支持通过串口、并口、红外端口来对远程机进行控制（不过，这里说的远程电脑，只能是有限距离范围内的电脑了）。传统的远程控制软件一般使用 NetBEUI、NetBIOS、IPX/SPX、TCP/IP 等协议来实现远程控制，不过，随着网络技术的发展，目前很多远程控制软件提供通过 Web 页面以 Java 技术来控制远程电脑，这样可以实现不同操作系统下的远程控制，从而完成我们所要介绍的远程监控与管理工作。

## 11.1.1 远程监控与管理原理

远程监控与管理依赖于远程控制技术，而该软件一般分为两个部分：一部分是客户端程序 Client，另一部分是服务器端程序 Server，在使用前需要将客户端程序安装到主控端电脑上，将服务器端程序安装到被控端电脑上。其控制的过程一般是先在主控端电脑上执行客户端程序，像一个普通的客户一样向被控端电脑中的服务器端程序发出信号，建立一个特殊的远程服务，然后通过这个远程服务，使用各种远程控制功能发送远程控制命令，控制被控端电脑中各种应用程序的运行，我们称这种远程控制方式为基于远程服务的远程控制。通过远程控制软件，我们可以进行很多方面的远程控制，包括获取目标电脑屏幕图像、窗口及进程列表；记录并提取远端键盘事件（击键序列，即监视远端键盘输入的内容）；可以打开、关闭目标电脑的任意目录并实现资源共享；提取拨号网络及普通程序的密码；激活、中止远端程序进程；管理远端电脑的文件和文件夹；关闭或者重新启动远端电脑中的操作系统；修改 Windows 注册表；通过远端电脑上、下载文件和捕获音频、视频信号等。前面我们所说的是一台电脑对一台电脑的情况，其实，基于远程服务的远程控制最适合的模式是一对多，即利用远程控制软件，我们可以使用一台电脑控制多台电脑，这就使得我们不必为办公室的每一台电脑都安装一个调制解调器，而只需要利用办公室局域网的优势就可以轻松实现远程多点控制了。在进行一台电脑对多台远端电脑进行控制时，远程控制软件类似一个局域网的网络管理员，而提供远程控制的远程终端服务就像办公室局域网的延伸。这种一对多的连接方式在节省了调制解调器的同时，还使得网络的接入更加安全可靠，网络管理员也更易于管理局域网上的每一台电脑。

## 11.1.2 远程监控与管理的主要应用范围

在通常情况下，远程监控与管理技术主要应用于以下两个方面。

### 1. 远程技术支持

通常，远距离的技术支持必须依赖技术人员和用户之间的电话交流来进行，这种交流既耗时又容易出错。许多用户对电脑知道得很少，然而当遇到问题时，他们必须向无法看到电脑屏幕的技术人员描述问题的症状，并且严格遵守技术人员的指示精确地描述屏幕上的内容，但是由于他们的电脑专业知识非常少，描述往往不得要领，说不到点子上，这就给技术人员判断故障制造了

非常大的障碍。即使技术人员明白了用户电脑的问题所在，在尝试解决问题时，技术人员可能会指导用户执行一系列复杂的命令，而这个过程对用户来说是十分困难的，因为技术人员要依靠自己的语言来“操纵”用户的鼠标和键盘简直是太难了，如果用户不能正确地遵照指示去做，问题可能会进一步恶化，电脑很可能会因为错误的操作导致系统的崩溃。这样一来，往往是技术人员要为十分简单的一个问题和用户说上十几分钟，甚至会专程跑到很远的用户那里帮助解决问题，而用户往往因为问题还没有解决，只好将电脑闲置不用，只等技术人员上门来解决问题。有了远程控制技术，技术人员就可以远程控制用户的电脑，就像直接操作本地电脑一样，只需要用户的简单帮助就可以得到该机器存在的问题的第一手材料，很快就可以找到问题的所在，并加以解决。

2．远程维护、监控和管理

网络管理员或者普通用户可以通过远程控制技术为远端的电脑安装和配置软件、下载并安装软件修补程序、配置应用程序和进行系统软件设置。在节假日或者是由于管理员在外地而通过远程方式对企业网络及其服务器进行维护和管理是属于该范畴的一个典型应用，下面所要讲述的也主要针对这种情况。

### 11.1.3 远程监控及管理的基本内容

远程监控及管理都包括哪些项目呢？通俗地讲基本上任何网络设备都可以通过远程管理的方式来设置。中小企业比较常见的远程管理项目主要有服务器、NAS 存储设备、路由器、交换机、防火墙、入侵检测系统等。

- 服务器管理：服务器是企业网络设备的主体，不管企业规模是大是小，都会拥有自己的服务器。而服务器的正常运行也是至关重要的，企业业务的正常开展全靠它了。
- 远程管理员工机及共享设备：实际上有的时候员工计算机也提供着某种服务，例如硬盘上存储着某些重要数据，而节日期间忽然要使用这些数据的话，如何才能从员工机中获取这些必要信息呢？除了连接到企业内部网络外，还需要学会远程管理员工计算机和一些办公共享设备。
- 远程管理路由器、交换机设备：路由器和交换机设备可以说是企业的核心网络设备了，它们负责互连工作。不过在实际使用过程中这些设备的设置会出现这样或那样的改动，根据时间和应用情况做适当修改。在节日期间如果需要对这些设备进行参数修改的话，我们就应该学会远程管理路由器、交换机设备。一般来说，我们在家中使用 telnet 命令就可以连接到路由交换设备上，不过这是建立在我们已经设置好远程访问参数的前提下。
- 远程管理其他设备：除了前面提到的各种网络设备外，像防火墙、入侵检测系统、NAS 等设备都可以进行远程管理，只要该设备具备图形化管理界面，那么我们就可以通过管理 IP 地址来远程管理。管理时只要注意保证使用的端口没有被封锁即可。

## 11.2 使用 Xmanager 3.0 实现 Linux 远程登录管理

通常大家所熟悉的 Putty 和 Secure SSH 软件基本上是都无法启动窗口服务的程序或进程，也无法达到远程桌面控制 Linux 的目的。在许多情况下，远程登录和桌面控制 Linux 也是非常必要和重要的管理工作。下面将介绍通过 Xmanager 远程桌面控制 Linux 的方法和技巧。

X是用在大多数UNIX系统中的图形支持系统。如果用户在自己的Linux机器上使用GNOME或者KDE的话，就正在使用X系统。它由X联盟（www.X.org）定义并维护。大多数的Linux用户使用的都是由XFree86项目（www.xfree86.org）提供的X Window系统的实现。xdm是一个显示管理器，提供了灵活的任务管理功能。然而xdm通常被认为是“GUI的登录屏幕，可以自动启动我的X任务”，用户会看到实际上它要更为强大。

xdm使用X联盟的X显示管理控制协议，即XDMCP，来和X服务器通信。它允许X服务器从运行xdm服务的服务器上获得会话服务。当使用xdm管理这些X任务的时候在设置上会有些复杂。但设置xdm可以得到本地的和其他服务器上的桌面。

## 11.2.1 配置Xmanager服务器端

由于Xmanager是共享软件，且运行于Windows控制端，所以获取非常容易，这里不再赘述。读者可以通过Internet轻松获得并安装，下面将以目前最新的Xmanager 3.0版本进行介绍。为了使用该软件，需要在Linux服务器上配置xdm，这里描述的配置允许任何XDMCP客户访问Linux服务器桌面环境。

为了使用Xmanager通过桌面方式监控和管理远程Linux，需要在远程被管理端的Linux上进行以下配置和文件修改工作。

（1）打开/etc/inittab文件，将runlevel 变为5，即id:5:initdefault: 如果原来就是5，则不用修改。

（2）对于Red Hat系列操作系统，打开/etc/X11/gdm/gdm.conf文件（对于Fedora系列是/etc/gdm/custom.conf文件）并找到[xdmcp]部分，将Enabled选项设为true或1；同时，要确保存在“Port=177”语句，因为177端口是用户要配置的XDMCP服务的监听端口。

（3）打开/etc/X11/xdm/xdm-config 文件，找到DisplayManager.requestPort: 0, 然后在前面加上字符“!”。

（4）打开/etc/X11/xdm/Xaccess文件，找到#*#any host can get a login window，将第一个“#”字符去掉。

（5）修改确保/etc/X11/xdm/Xservers的属性为444，/etc/X11/xdm/Xsetup_0的属性为755。当然，有一些Linux操作系统，比如在Red Hat Linux中，用户可以看到这两个文件默认的属性就是444和775，因此就不需要修改。

（6）如果用户的Linux机器配置有防火墙，为防止防火墙将通过177端口（即xdmcp服务）的数据过滤，用户必须加上以下防火墙规则。

```
#iptables -A INPUT -p udp -s 0/0 -d 0/0 --dport 177 -j ACCEPT
```

（7）重新启动Linux即可。

## 11.2.2 配置Xmanager客户端

另外，用户还需要在Windows管理和控制端安装软件Xmanager，在本文中采用Xmanager 3.0版本，安装过程非常简单，在此不再赘述。安装好该软件后，可以根据以下配置来完成监控和管理工作。

（1）启动Xbrowser，选择File菜单中的New Session Wizard命令，如图11-1所示。

（2）系统弹出New Session Wizard对话框，选中XDMCP单选按钮，如图11-2所示。

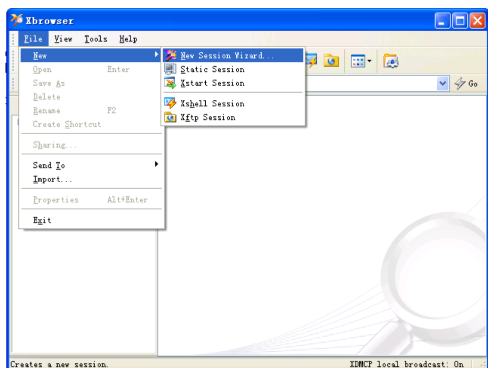



图 11-1　Xbrowser 窗口

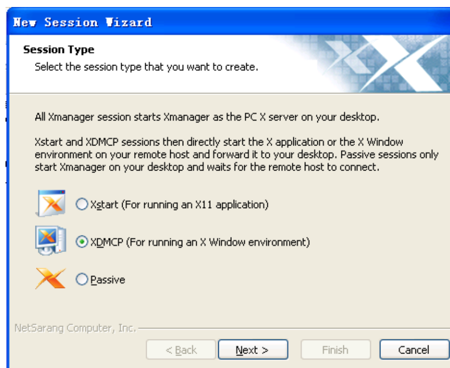



图 11-2　New Session Wizard 对话框

（3）将 Host 设为 Linux 机器的 IP，为 192.168.10.88，Port Number 为 177，如图 11-3 所示。

（4）设置完成后，将在 Xbrowser 中出现配置好的机器的图标，如图 11-4 所示。

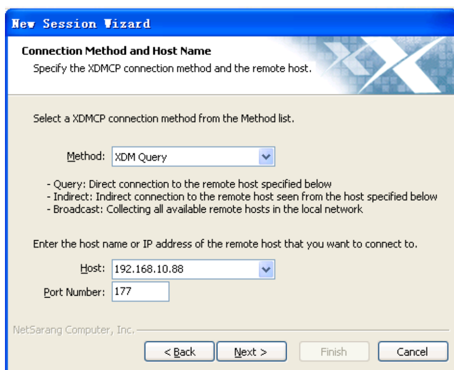



图 11-3　设置远程 Linux 机器的 IP 地址和端口

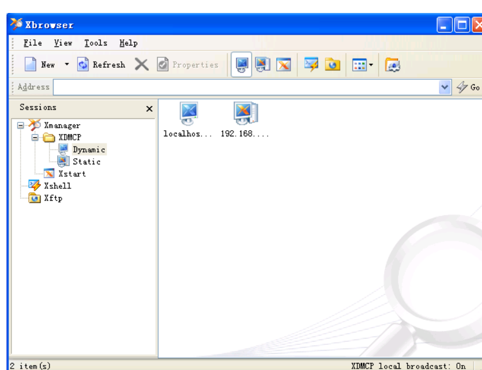



图 11-4　设置完成

（5）配置完成后，单击图 11-4 中的相应图标进入远程系统后则能根据需求进行远程监控和管理工作，如图 11-5 所示。

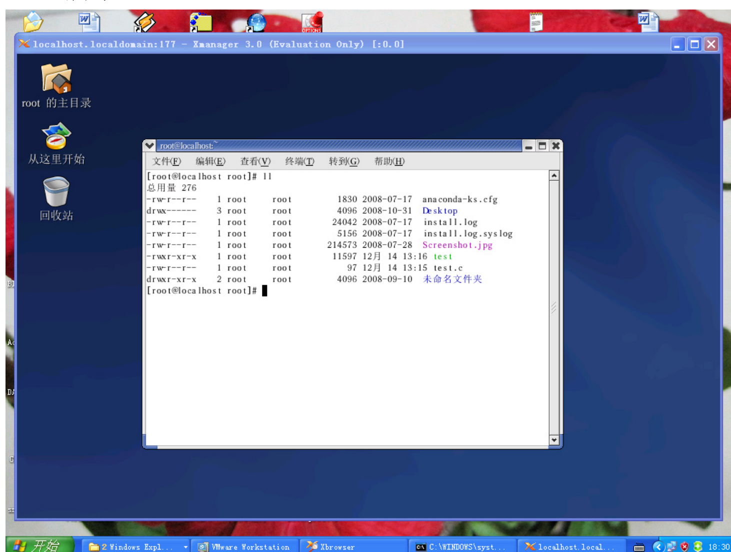



图 11-5　使用 Xmanager 远程管理 Linux

## 11.3 使用 VNC 实现 Linux 远程管理

### 11.3.1 VNC 简介

在开源领域，远程遥控技术的代表就是 VNC 了。VNC（Virtual Network Computer，虚拟网络计算机）是一套由 AT&T 实验室开发的可操控远程计算机的软件，该软件是开源的。也就是说，任何人都可以免费取得并不受限制地使用这款软件。根据主控端与被控端的不同，VNC 软件可以分为两个部分，分别为 VNC Server 与 VNC Viewer。前者是安装在被控制端上，而后者被安装在主控端上。VNC 软件不仅是开源的，而且是跨平台的。也就是说，其不仅在 Linux 操作系统上可以使用，而且也支持 Windows 操作系统。因此有不少系统管理员，他们可能使用的是微软的操作系统，也喜欢使用这个 VNC 来作为远程管理 Linux 服务器或者客户端的工具。

整个 VNC 运行的工作流程如下。

（1）VNC 客户端通过浏览器或 VNC Viewer 连接至 VNC Server。

（2）VNC Server 传送一对话窗口至客户端，要求输入连接密码，以及存取的 VNC Server 显示装置。

（3）在客户端输入联机密码后，VNC Server 验证客户端是否具有存取权限。

（4）若是客户端通过 VNC Server 验证，客户端即要求 VNC Server 显示桌面环境。

（5）VNC Server 通过 X Protocol 要求 X Server 将画面显示控制权交由 VNC Server 负责。

（6）VNC Server 将来由 X Server 的桌面环境利用 VNC 通信协议送至客户端，并且允许客户端控制 VNC Server 的桌面环境及输入装置。

### 11.3.2 启动 VNC 服务器

利用 VNC 软件实现远程控制的基本原理是主控端利用 VNC 客户端发起连接请求，被控端同意后即可建立远程控制。此时主控端就可以远程操控被控端。因此要利用 VNC 软件来远程操控 Linux 操作系统的话，必须先在 Linux 操作系统上启动 VNC 服务器软件。否则是无法建立 VNC 连接的。不过在大部分的 Linux 操作系统中，如红帽子的 Linux 系统，一般默认都会安装有 VNC 服务器的。不过其出于安全的考虑，一般都是关闭的。如果系统管理员想利用 VNC 来实现远程操控的话，就需要在 Linux 操作系统上启动 VNC 服务器。

在 Linux 操作系统的命令行下，系统管理员可以输入 vncserver 命令来启动 VNC 服务器。在启动的过程中为了安全，操作系统会提示系统管理员输入 VNC 连接的密码。系统管理员最好能够在这里输入比较复杂的密码，如英文字符与数字结合的密码，以增加破译的难度。由于建立 VNC 连接后，主控端可以像操作自己的电脑那样来操作被控端，所以这个密码将是保障其安全的最后屏障。启动成功后如图 11-6 所示。

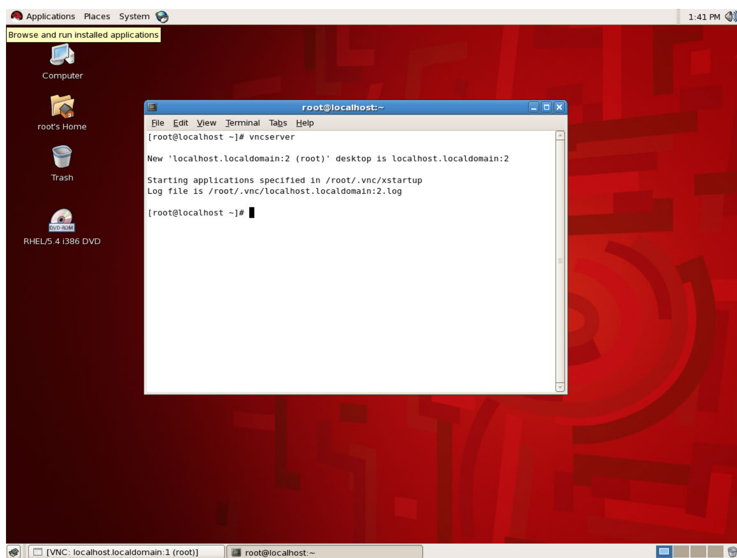



图 11-6　启动 VNC Server

密码配置完成后，Linux 操作系统最后还会提示 VNC 连接的地址。如上面所示，最后系统管理员可以使用 localhost.localdomain 来远程操控 Linux 操作系统。以后在 VNC 客户端上只要输入这个网络地址，就可以连接到 VNC 服务器上。如果后续需要更改 VNC 的连接密码，需要使用 vncpasswd 来进行更改。注意不是 passwd。这和更改用户密码的命令不同。一般情况下，只要正常显示了其网络地址，那么这个 VNC 服务就是正常启动了。

另外为了安全，中断 VNC 服务器之后，最好在服务器上能够及时关闭 VNC 应用服务器。关闭的命令如上，只需要运行以下命令即可：

```
#vncserver -kill :1
```

其中最后的 1 表示之前启动的窗口编号。系统管理员应该养成一个习惯，即当某个服务启动后，要及时关闭它。多启动一个服务，就多给黑客一个攻击的机会。特别是将操作系统当作服务器来使用的时候，这个习惯能够在很大程度上提高服务器系统的安全性。其实不仅仅是 VNC 服务器如此，像其他的 Telnet 服务等也要养成这个习惯。一般来说，Linux 操作系统默认不启动的服务，系统管理员在启动它们之后，最后都要及时关闭。

不过如果企业的布局比较大，如从系统管理员的办公室到 Linux 操作系统客户端那边需要走半个小时。此时为了管理的方便，如果对方操作系统只是用来作普通的客户端的话，那么就可以让 Linux 操作系统在启动时自动启动 VNC 服务器，以方便在遇到问题时，系统管理员能够及时连接上 VNC 服务器进行远程故障排除或者远程协助。由于客户端的安全性要求不怎么严格，所以在管理便利方面上可以做出一定程度的妥协。但是如果操作系统是作为服务器的话，那么开机自动启动 VNC 服务器类似的操作，系统管理员需要谨慎。如果系统管理员确定需要开机时自动启动 VNC 服务器，则可以通过 ntsysv 服务来定义。即只需要在命令行状态下，输入命令 ntsysv，然后选中 vncserver 条目（按空格键选择），即设置了启动 VNC 服务器。然后需要修改 /etc/sysconfig/vncservers 配置文件。找到这个文件中的 VNCSERVER=“1:root”条目，默认情况下操作系统是将这一行注释掉的。系统管理员只需要将前面的注释符号去掉即可。如此设置后，当操作系统在下次启动后就会自动启用 VNC 服务器。这样系统管理员就可以远程来控制 Linux 操作系统，进行软件安装、系统配置、远程协作等操作。

### 11.3.3 使用 VNC Viewer 实现 Linux 远程管理

在配置好 VNC Server 后，可以使用 VNC Viewer 来实现 Linux 的远程登录和管理了。在 Linux 下面已经自带了该客户端程序，用户可以打开“应用程序”菜单，在弹出的级联菜单中选择“VNC Viewer”命令，如图 11-7 所示。

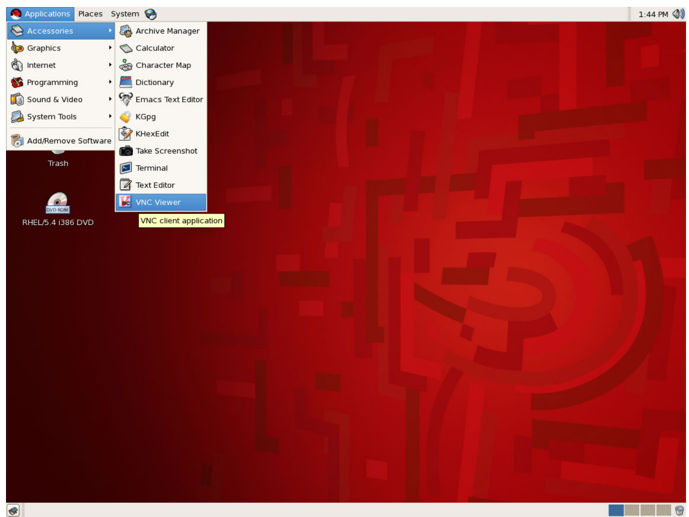



图 11-7 使用 VNC Viewer

系统弹出如图 11-8 所示的 VNC Server 登录对话框，用户需要指定待登录的服务器地址。

输入地址后，VNC Viewer 连接上 VNC Server，Server 需要对 VNC 客户端进行身份验证，弹出如图 11-9 所示的对话框，用户输入之前设置的密码即可。

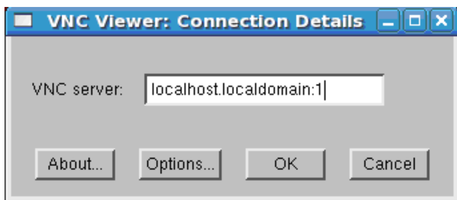



图 11-8 输入 VNC Server 地址

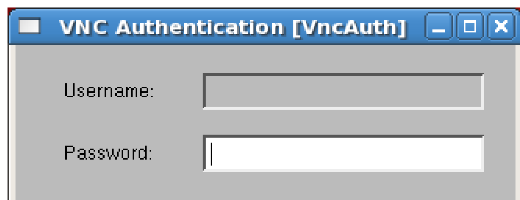



图 11-9 输入密码

待客户端通过服务器端的验证后，则会进入如图 11-10 所示的远程登录界面，用户即可在图形界面下简单、方便地对远程的 Linux 进行访问和管理了。

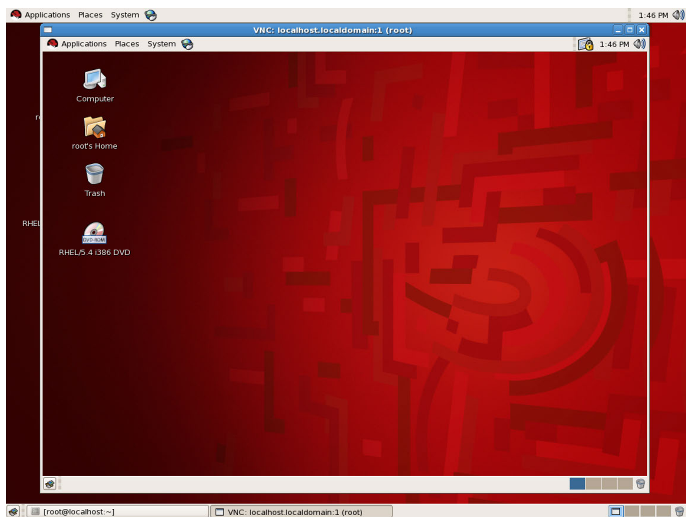



图 11-10 成功使用 VNC Server

## 11.3.4 使用 SSH+VNC 实现安全的 Linux 远程桌面管理

VNC 可以对数据进行压缩，使得传输的数据量比直接用 SSH 加密的小，但缺点是有一到两次机会让同一网段的计算机用 sniffer 窃听到用户名和密码。认证之后的数据可以进行加密传输，所以使用过程中若经过配置，则是安全的，否则传输内容不能保证完全保密。必要时可以通过 SSH 进行加密端口映射来保证传输用户名和密码的时候也是加密的，这一操作占用的额外带宽是极少的。

至于 SSH 保护 VNC 的安全，需要使用 SSH 的端口转发功能。一般在客户端使用的是 Linux 的时候，可以先用 SSH 建立连接，语法上应当添加“-L 本地端口:本地地址:远程端口 远程地址”这一附加参数，比如说本地是 X.Y.Z.W，服务器是 A.B.C.D，要转发的端口本地的是 5901，远程的也是 5901，那么命令应当是：

```
ssh -L 5901:X.Y.Z.W:5901 A.B.C.D
```

其他的参数多数可以同时使用。执行完毕后就已经创建了服务器 5901 端口和本地 5901 端口的加密隧道。假设要连接的服务器上运行的 VNC 桌面号是 2，则继续执行以下命令。

```
vncviewer A.B.C.D:2
```

这样打开的 VNC 窗口所有数据都经过了 SSH 的加密。

由于一般管理情况下多采用 SSH 的 Windows 客户端，也就是 SSH Secure Shell，所以下面介绍如何配置 Windows 下的 SSH Secure Shell 来配合 VNC 实现安全的 Linux 远程桌面管理。

首先，在 SSH Secure Shell 的主界面上选择 Settings 下的 Tunneling，如图 11-11 所示。

然后，选择添加一个配置，如图 11-12 所示，其中 Listen 是本机端口，Destination 是远程地址和端口，Display 可以设定自己的描述。

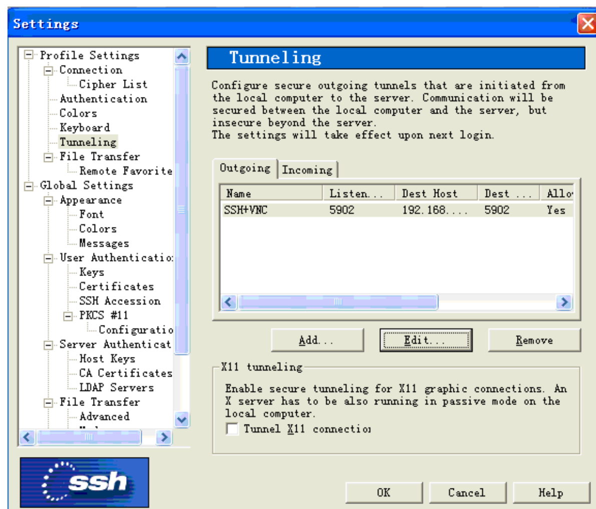



图 11-11 为 SSH Secure Shell 设定 Tunneling

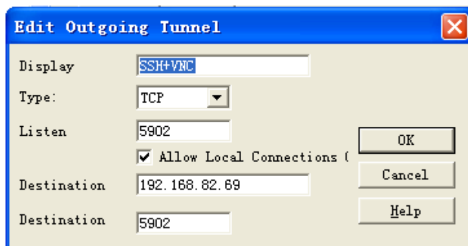



图 11-12 Tunneling 设定的具体选项

最后，再运行 vncviewer 来连接即可。特别值得注意的是：此处 Server 的地址不是需要连接的 VNC Server 的地址，而是前面设定的本机的 SSH 侦听端口的地址，因为所有通往 VNC Server 的流量都需要先经过本机的 SSH 进行转发，如图 11-13 所示，一般设定为 localhost 加端口即可，

该端口就是在图 11-12 中设定的 SSH 的侦听端口。

图 11-14 清晰地给出了使用 Wireshark 对上述通信的流量进行抓包的结果，可以看到，从客户端去往服务器端的 VNC 流量，均采用 SSH 协议进行了加密传输，因此针对传统的未经过 SSH 加密处理的 VNC 通信来说要安全很多，能够有效地避免窃听和中间人攻击。

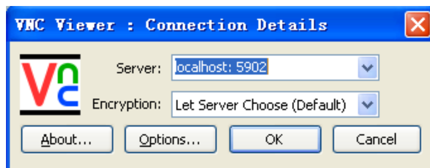



图 11-13　使用 Windows 中的 VNC Viewer 连接 Linux 端的 VNC Server

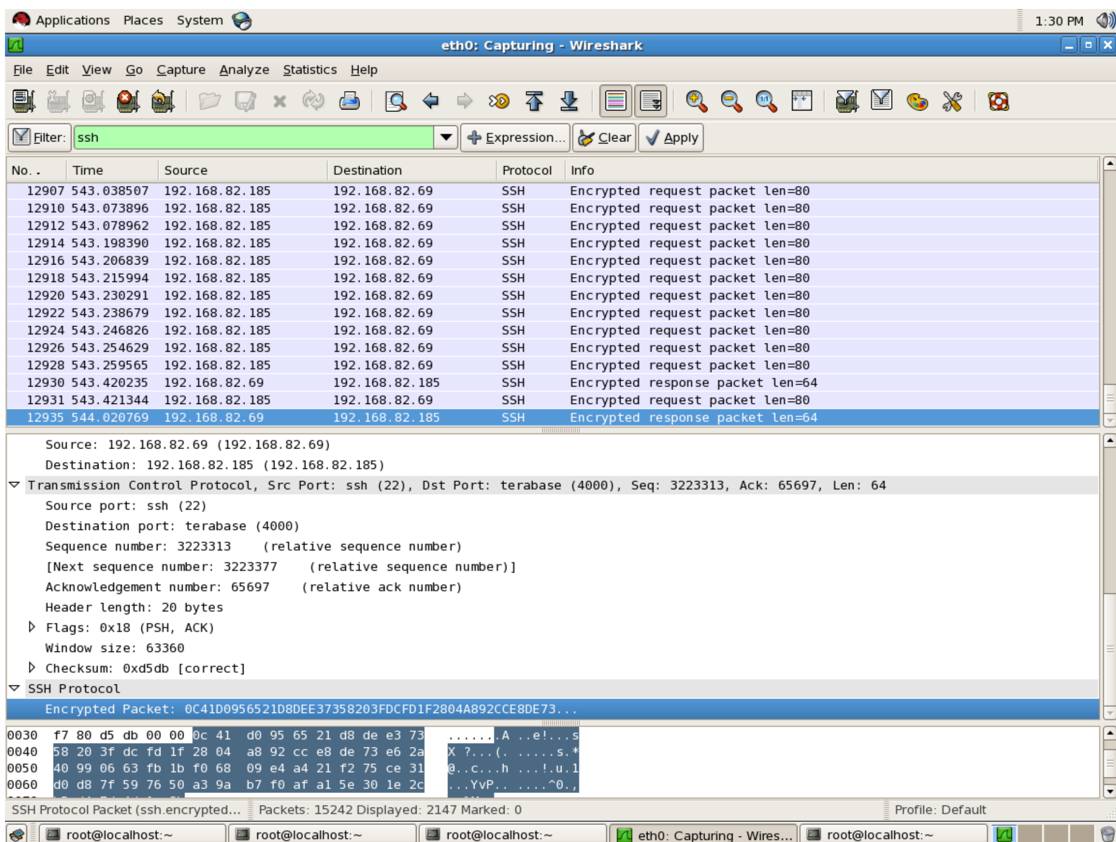



图 11-14　针对上述通信流程的抓包结果显示

# 第 12 章

# 举重若轻：企业网络流量安全管理

## 本章导读

近十几年来，互联网已经成为越来越重要的需求。据统计，互联网目前已成为人类社会最重要的信息基础设施，占人类信息交流的 80%。它对社会进步、经济发展和国家安全具有重大战略意义。在这种大背景下，面对日益复杂的网络联机及逐渐增加的网络流量，系统和网络管理者必须花费更多时间和精力来了解这些网络设备的运作状况，以维持一个系统的正常运作。

一般来说，网络管理者所需要了解的是各个网段的使用情形、频宽的使用率、网络问题的瓶颈发生于何处。当网络问题发生时，必须能够很快地分析和判断出问题的发生原因，可能是线路问题、网络设备问题或者是路由器的设定问题。对网络流量进行有效的管理是最大化发挥网络效能的首要因素。那么，500 强企业管理网络流量的时候应该通过什么样的依据管理，通过什么手段有效地识别流量，并分析和管理呢？本章将针对 Linux 网络，介绍网络流量管理的主要技术和工具。

# 12.1 网络流量管理简介

一般来说，我们可以将网络流量管理大致分为以下几个大的范畴。

## 12.1.1 流量识别

流量识别，也叫业务识别|（Application Awareness），它是伴随着网络业务的蓬勃发展而出现的一个新的概念，通过对业务流量从数据链路层到应用层的报文深度检查分析，依据协议类型、端口号、特征字符串和流量行为特征等参数，获取业务类型、业务状态、业务内容和用户行为等信息，并进行分类统计和存储。业务识别的基本目的是帮助网络管理者获得网络层之上的业务层流量信息，如业务类型、业务状态、业务分布、业务流量流向等。业务识别是一个相对复杂的过程，需要多个功能模块的协同工作。业务识别的工作过程简单地描述如下。

- 识别处理模块采用多通道识别处理，通过对网络流量的源/目的 IP 地址和源/目的端口号的 Hash 算法，将网络流量均匀地分配到多个处理通道中。
- 多处理通道并行执行网络流量的深度报文检查，获取网络流量的特征信息，并与业务识别特征库中的特征进行比对。
- 将匹配结果送往识别处理模块，并标识特定网络流量。如果存在多个匹配结果，选取优先级较高的匹配结果进行标识。特定网络流量一经识别确定，该网络流量的后续连接将不再进行深度报文检查，直接将其网络层和传输层信息与已知识别结果进行比对，提高执行效率。
- 识别处理模块将网络流量的业务识别结果存储到识别结果存储模块中，为网络流量的统计分析提供依据。
- 统计分析模块从识别结果存储模块中读取相关信息，并以曲线、饼图、柱状图或者文本的方式将识别结果信息显示，或以文件的形式输出。
- 在结果存储模块中保存的识别结果信息会输出到网络流量管理功能区，为实施网络流量管理提供依据。

目前常用、典型的业务识别技术就是我们所熟知的 DPI 技术和 DFI 技术。

### 1. DPI 技术

DPI 是深度报文检测（Deep Packet Inspection）的简称，是一种典型的业务识别技术。DPI 技术之所以称为“深度”的检测技术，是相对于传统的检测技术而言的。传统的流量检测技术仅获取那些寄存在数据包网络层和传输层协议头中的基本信息，包括源/目的 IP 地址、源/目的传输层端口号、协议号，以及底层的连接状态等。通过这些参数很难获得足够多的业务应用信息。对于当前 P2P 应用、VoIP 应用、IPTV 应用被广泛开展的情况，传统的流量检测技术已经不能满足网络流量管理的需要了。

DPI 技术对传统的流量检测技术进行了“深度”扩展，在获取数据包基本信息的同时，对多个相关数据包的应用层协议头和协议负荷进行扫描，获取寄存在应用层中的特征信息，对网络流量进行精细的检查、监控和分析。

DPI 技术通常采用以下数据包分析方法。

- 传输层端口分析。许多应用使用默认的传输层端口号，例如 HTTP 协议使用 80 端口。

- 特征字匹配分析。一些应用在应用层协议头，或者应用层负荷中的特定位置中包含特征字段，通过特征字段的识别实现数据包检查、监控和分析。
- 通信交互过程分析。对多个会话的事务交互过程进行监控分析，包括包长度、发送的包数目等，实现对网络业务的检查、监控和分析。

2．DFI 技术

DFI 是深度流行为检测（Deep Flow Inspection）的简称，也是一种典型的业务识别技术。DFI 技术是相对于 DPI 技术提出的，是为了解决 DPI 技术的执行效率、加密流量识别和频繁升级等问题而出现的。DFI 更关注于网络流量特征的通用性，因此，DFI 技术并不对网络流量进行深度的报文检测，而仅通过对网络流量的状态、网络层和传输层信息、业务流持续时间、平均流速率、字节长度分布等参数的统计分析，来获取业务类型、业务状态。

两种技术的设计基本目标都是为了实现业务识别，但是两者在实现的着眼点和技术细节方面还是存在着较大区别的。从两种技术的对比情况看，两者互有优势，也互有短处，DPI 技术适用于需要精细和准确识别、精细管理的环境，而 DFI 技术适用于需要高效识别，粗放管理的环境。

### 12.1.2　流量统计分析

网络流量管理的基本目标是了解网络、业务和用户资源的使用情况，找到性能瓶颈并进行精细化管理，对用户行为进行分析和控制，以及对信息安全防护。

在网络中多个层面的网络设备中集成业务识别功能，如骨干节点之间、服务提供商互联节点之间、国际出口、城域网出口和城域网接入层等，或者单独部署具备业务识别功能的网络设备对网络流量进行统计分析和趋势判断。通过流量统计分析，网络管理者能够知道当前网络中的业务流量的类型、带宽、时间和空间分布、流向等信息。

### 12.1.3　流量限制

将流量限制能力添加到网络流量管理中，能够帮助网络管理者对网络资源和业务资源进行带宽控制和资源调度。具备流量限制能力的网络流量管理将具备 P2P 应用的管理能力，通过对 P2P 流量的抑制来提升传统数据业务的用户体验度。具备流量限制能力的网络流量管理还能够对严重影响业务运营者收入的未经许可的业务进行抑制。通过对 VoIP 信令流量和媒体流量的关联检测和统计分析，以及截断媒体数据包、伪装信令报文等方式对 VoIP 业务进行流量管理。通过综合使用网络层、传输层和应用层检测技术，对未经许可的宽带私接用户采取中断连接、主动告警、分时控制等多种管理动作，实现对未经许可的宽带私接的流量管理。流量限制还能够帮助网络流量管理实现业务资源的调度，流量限制能够获得业务资源使用、业务状态的实时情况。当某一业务服务器负载较大时，可以进行全局的业务资源负载均衡，平均地承担业务请求；同时也能够对用户的业务请求进行调度，决定是否继续响应用户新的业务请求，或者根据用户的优先级，优先响应高优先级用户的业务请求，提升业务运营效率。

### 12.1.4　其他方面

对于网络流量管理来说，在网络中的多个层面的网络设备中集成业务识别功能，或者单独部署具备业务识别功能的网络设备，和防火墙等网络安全设备协同构建一个主动的安全威胁防御体

系，提升整个网络的安全防护能力。

具备业务识别能力的网络流量管理具有主动的流量特征识别分析能力，能够主动地发现诸如DDoS攻击、病毒和木马等异常流量，较好地弥补其他网络安全设备，如防火墙、入侵防护系统（IPS）和统一威胁管理（UTM）等的不足，提升其主动发现安全威胁的能力，并能够及时地向其他网络安全设备发出告警，从安全威胁源头开始就进行主动的防御。

此外，具备业务识别能力的网络流量管理还能够获取并保存网络流量的网络层信息（例如，源/目的 IP 地址、用户标识 ID 等信息），通过这些信息，网络管理者能够进行有效的安全威胁的溯源定位。

## 12.2 需要管理的常见网络流量

当前随着网络应用的不断丰富和发展，网络流量也随之变得复杂和种类繁多起来。下面给出几种最为常见的网络流量，以方便网络管理员在实际的工作中着重对它们进行监测和管理。

- HTTP 流量。HTTP 是互联网 TCP/IP 协议族中的一种应用层协议，被广泛应用于网页流量、网络下载等。HTTP 协议正在取代传统文件下载的主要应用层协议 FTP，此外基于 HTTP 的网页版邮箱也有取代传统的 POP3 协议的趋势。根据国外媒体报道，最近发布的一个研究报告指出，随着 YouTube 等视频共享网站的拉动，传统的 HTTP 协议的网络流量在过去四年里首次超过了 P2P 应用的流量。基于 HTTP 协议的互联网流量已经占据了全部流量的 46%；排名第二的是 P2P 应用的流量，其占据了 37%的比例；排在后面的流量分别属于新闻组（9%）；非 HTTP 协议的视频流媒体（3%）；网络游戏（2%）；以及 VOIP 网络电话（1%）。
- FTP 流量。FTP 是互联网中一种应用非常广泛的服务，用户通过它来从服务器获取需要的文档、资料、音频、视频等。从互联网出现的开始，它就一直是用户使用频率最高的应用服务之一，重要性仅次于 HTTP 和 SMTP。而随着 P2P 应用的出现，其重要性地位虽然有所降低，但是仍然是用户下载文件不可替代的重要应用和途径之一。
- SMTP 流量。电子邮件是整个互联网业务重要的组成部分。据统计，四分之三以上的用户上网的主要目的是收发邮件，每天有数十亿封电子邮件在全球传递。电子邮件已成为网络用户不可或缺的工具。并且，电子邮件的廉价和操作简便在给人们带来巨大便利的同时，也诱使有些人将它作为大量散发自己信息的工具，最终导致互联网世界中垃圾邮件的泛滥。垃圾邮件极大地消耗了网络资源，并给人们带来了极大的不便。据中国互联网协会（ISC）2005 年第一次反垃圾邮件状况调查显示，中国邮件用户 2005 年 4 月平均每人每天收到邮件 16.8 封，占收到邮件总数的 60.87%。因此，SMTP 流量目前占据相当大的互联网流量比重。
- VoIP 流量。2006 年全球 IP 电话用户从 1030 万增长到 1870 万，增幅达 83%。2007 年 VoIP 通话量将达到全部通话量的 75%。数据显示，PC2Phone 的 IP 电话付费用户数量超过 470 万人，算上运营商 IP 电话服务的预定用户的话，这一数字将达到 2400 万。因此，互联网上 VoIP 的流量也是非常值得管理员关注的。
- P2P 流量。报告指出，目前网络带宽“消费大户”是 P2P 文件共享，在中东占据了 49%，

东欧地区占据了 84%。从全球来看，晚上时段的网络带宽有 95%被 P2P 占据。在所有 P2P 工具中，BitTorrent 最受欢迎，在南欧地区，电驴处于主导地位。在 P2P 内容方面并未和去年发生变化，主要还是视频，最受欢迎的是最新上映的电影、色情电影和音乐。其中在中东，电子书在 P2P 内容中比例较高，计算机游戏在南欧地区比例较高。

- Streaming 流量。随着诸如 PPLive、PPStream 等视频软件的出现，视频直播和点播成为广大互联网用户观看节目和网上娱乐的最佳生活方式，因此其流量也在不断的剧增当中。并且，随着 P2P 技术的发展，P2P Streaming 也成为今后互联网中一项非常重要的应用。

## 12.3 网络流量捕捉：图形化工具 Wireshark

### 12.3.1 Wireshark 简介

Ethereal 是一个开放源码的网络分析系统，也是目前最好的开放源码的网络协议分析器，支持 Linux 和 Windows 平台。Ethereal 起初由 Gerald Combs 开发，随后由一个松散的 Ethereal 团队组织进行维护开发。它目前所提供的强大的协议分析功能完全可以媲美商业的网络分析系统，自从 1998 年发布最早的 0.2 版本至今，大量的志愿者为 Ethereal 添加新的协议解析器，如今 Ethereal 已经支持五百多种协议解析。另外，网络分析系统首先依赖于一套捕捉网络数据包的函数库，这套函数库工作在网络分析系统模块的最底层，作用是从网卡取得数据包或者根据过滤规则取出数据包的子集，再转交给上层分析模块。从协议上来说，这套函数库将一个数据包从链路层接收，至少将其还原至传输层以上，以供上层分析。6 月 8 号，Ethereal 的作者 Gerald Combs 宣布离开 NIS 的消息，因而 Ethereal 现改名为 Wireshark。

在 Linux 系统中，1992 年 Lawrence Berkeley Lab 的 Steven McCanne 和 Van Jacobson 提出了包过滤器的一种实现——BPF（BSD Packet Filter）。Libpcap 是一个基于 BPF 的开放源码的捕包函数库。现有的大部分 Linux 捕包系统都是基于这套函数库或者是在它的基础上做一些针对性的改进。在 Windows 系统中，意大利人 Fulvio Risso 和 Loris Degioanni 提出并实现了 Winpcap 函数库，作者称之为 NPF。由于 NPF 的主要思想就是来源于 BPF，它的设计目标就是为 Windows 系统提供一个功能强大的开发式数据包捕获平台，希望在 Linux 系统中的网络分析工具经过简单编译以后也可以移植到 Windows 中，因此这两种捕包架构是非常现实的。就实现来说，提供的函数调用接口也是一致的。Ethereal 网络分析系统也需要一个底层的抓包平台，在 Linux 中是采用 Libpcap 函数库抓包，在 Windows 系统中采用 Winpcap 函数库抓包。

### 12.3.2 层次化的数据包协议分析方法

使用捕包函数捕捉数据包后需要进行协议分析和协议还原工作。由于 OSI 的 7 层协议模型，其协议数据是从上到下封装后发送的。对于协议分析需要从下至上进行。首先对网络层的协议识别后进行组包还原，然后脱去网络层协议头，将里面的数据交给传输层分析，这样一直进行下去直到应用层。由于网络协议种类很多，就 Ethereal 所识别的 500 多种协议来说，为了使协议和协议间层次关系明显，从而对数据流里的各个层次的协议能够逐层处理，Ethereal 系统采用了协议树的方式。图 12-1 所示就是一个简单的协议树。如果协议 A 的所有数据都是封装在协议 B 里的，

那么这个协议 A 就是协议 B 的另外一个协议的儿子节点（比如图 12-1 中的 TCP 和 UDP 协议就是 IP 协议的儿子节点）。我们将最底层的无结构数据流作为根节点。则具有相同父节点的协议称为兄弟节点。那么这些拥有同样父协议兄弟节点协议如何来区分呢？Ethereal 系统采用协议的特征字来识别。每个协议会注册自己的特征字，这些特征字给自己的子节点协议提供可以互相区分开来的标识，比如 TCP 协议的 port 字段注册后，Tcp.port=21 就可以认为是 FTP 协议，特征字可以是协议规范定义的任何一个字段，比如 IP 协议就可以定义 Port 字段为一个特征字。

在 Ethereal 中注册一个协议解析器首先要指出它的父协议是什么，另外还要指出自己区别于父节点下的兄弟节点协议的特征，比如 FTP 协议。在 Ethereal 中它的父节点是 TCP 协议，它的特征就是 TCP 协议的 Port 字段为 21。这样当一个端口为 21 的 TCP 数据流来到时，首先由 TCP 协议注册的解析模块处理，处理完之后通过查找协议树找到自己协议下面的子协议，判断应该由哪个子协议来执行，找到正确的子协议后，就转交给 FTP 注册的解析模块处理，这样由根节点开始一层层解析下去。

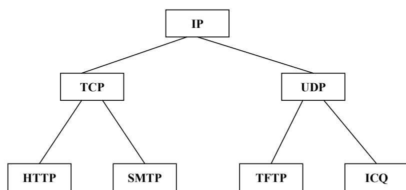



图 12-1　简单协议树

由于采用了协议树加特征字的设计，这个系统在协议解析上有了很强的扩展性，增加一个协议解析器只需要将解析函数挂到协议树的相应节点上即可。

## 12.3.3　基于插件技术的协议分析器

所谓插件技术，就是在程序的设计开发过程中，将整个应用程序分成宿主程序和插件两个部分，宿主程序与插件能够相互通信，并且，在宿主程序不变的情况下，可以通过增减插件或修改插件来调整应用程序的功能。运用插件技术可以开发出伸缩性良好、便于维护的应用程序。其著名的应用实例有：媒体播放器 Winamp、微软的网络浏览器 IE 等。

由于现在网络协议种类繁多，为了可以随时增加新的协议分析器，一般的协议分析器都采用插件技术，这样如果对一个新的协议分析只需要开发编写这个协议分析器并调用注册函数在系统注册就可以使用了。通过增加插件使程序有很强的可扩展性，各个功能模块内聚。

## 12.3.4　安装 Wireshark

Wireshark 可以在 http://www.wireshark.org/download.html 上下载（其主页参见图 12-2），该软件有极其方便和友好的图形用户界面，并且能够使得用户通过图形界面的配置和选择，针对多块网卡、多个协议进行显示，效果非常好。目前最新版本为：Wireshark 1.0.7。安装该软件请按照以下步骤进行。




图 12-2　Wireshark 网站主页

（1）将下载的最新版本软件拷贝到临时文件夹：

```
# cp wireshark-1.0.7.tar.gz /usr/local/src/
```

（2）切换到临时文件夹目录：

```
# cd /usr/local/src/
```

（3）解压缩文件：

```
# tar -xvf wireshark-1.0.7.tar.gz
```

（4）另外，同 Tcpdump 一样，在编译 Ethereal 之前应先确定已经安装 pcap 库（Libpcap），这是编译 Wireshark 时所必需的。如果该库已经安装，就可以执行下面的命令来编译并安装 Wireshark：

```
# cd wireshark-1.0.7
# ./configure
# make
# make install
```

当编译并安装好 Wireshark 后，就可以执行 Wireshark 命令来启动 Wireshark。

## 12.3.5　使用 Wireshark

### 1. 捕包选项

抓包是进行协议分析的第一步骤，在 Wireshark 软件主界面（参见图 12-3）中，有几个抓包的相关选项需要特别注意，如图 12-4 和图 12-5 所示。

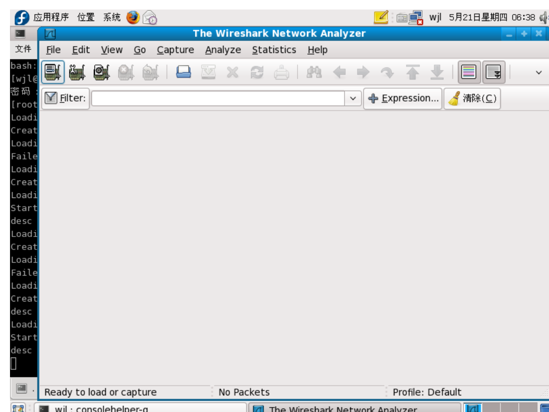


图 12-3　Wireshark 主界面

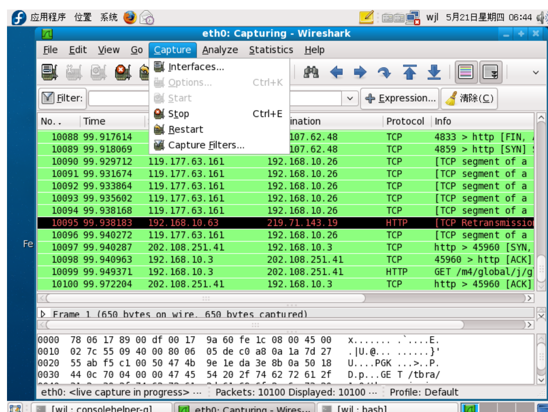


图 12-4　Wireshark Capture 选项

- Interface：指定在哪个接口（网卡）上抓包。一般情况下都是单网卡，所以使用默认的就可以。
- Limit each packet：限制每个包的大小，默认情况下不限制。
- Capture packets in promiscuous mode：确定是否打开混杂模式。如果打开，抓取所有的数据包。一般情况下只需要监听本机收到或者发出的包，因此应该关闭该选项。

- Filter：过滤器。只抓取满足过滤规则的包。
- File：如果需要将抓到的包写到文件中，在这里输入文件名称。
- use ring buffer：是否使用循环缓冲。默认情况下不使用，即一直抓包。值得注意的是：循环缓冲只有在写文件的时候才有效。如果使用了循环缓冲，还需要设置文件的数目，文件多大时回卷。
- 其他项的选择默认的就可以了。

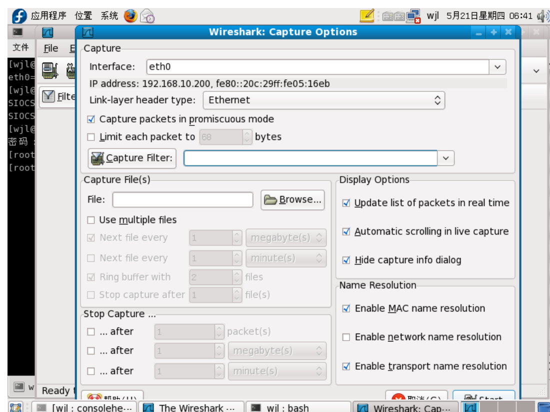


图 12-5　Wireshark 捕包选项设置

## 2．协议过滤（Filter）选项

由于网络中的协议五花八门，HTTP、FTP、ARP、ICMP 等协议等都在抓包的范围之列，因而给协议分析工作带来了一些麻烦。为了对特定的协议进行分析统计，则可以使用 Ethereal 提供的 Filter 选项来对数据报文进行过滤，对特定的协议进行提取，再进行分析。图 12-6 给出了使用 Capture 菜单下的 Capture Filter 命令进行过滤设置的示意，该软件已经提供了许多协议的 Filter 供用户选择使用，而用户也可以自行添加；图 12-7 则显示了通过在工具栏的 Filter 编辑框内输入协议名称对已捕捉的包进行过滤的结果，图中显示了过滤后所得 HTTP 协议的数据包情况。用户也可以通过输入 FTP、ARP 或者 TCP 等字段来获取相应的协议内容。

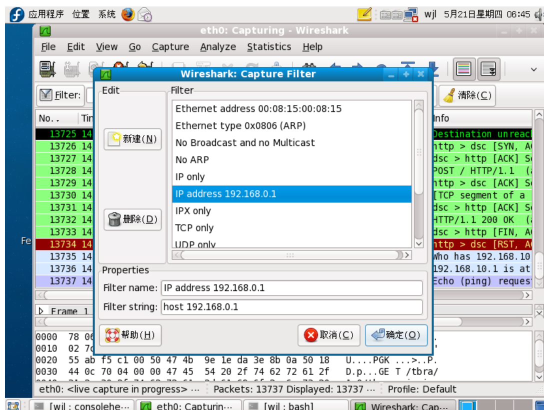


图 12-6　下拉菜单协议过滤选项

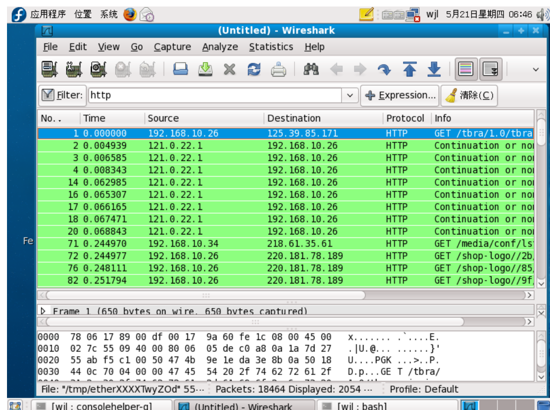


图 12-7　工具栏协议过滤选项

> **注意**　图 12-6 和图 12-7 所示的设置方法的不同在于：前者设置后该工具只对选取的协议进行捕包；而后者则是在所有捕获的数据包（包括各种协议）中选择用户设定的协议进行提取和显示。

## 3．统计（statistics）选项

统计选项用于对过滤后的各协议类型进行统计，从而从宏观上对网络中的流量进行统计分析和全局把握。图 12-8 给出了操作的流程，图 12-9 给出了显示结果。

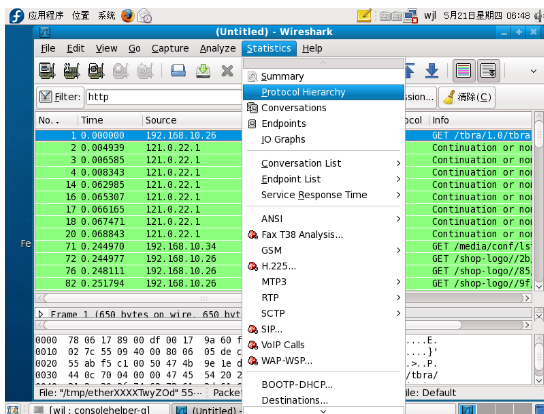


图 12-8　统计选项选择

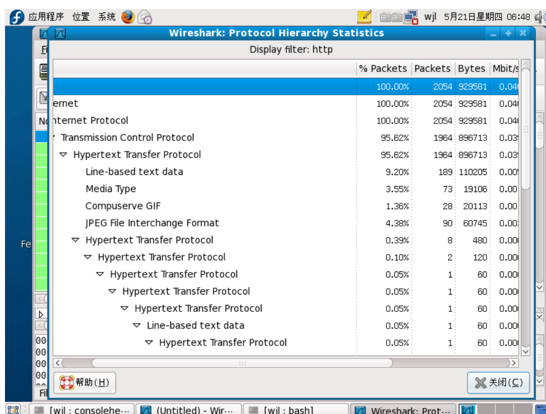


图 12-9　统计结果

## 12.4　网络流量捕捉：命令行工具 tcpdump

### 12.4.1　tcpdump 简介

tcpdump 可以将网络中传送的数据包的“头”完全截获下来提供分析。它支持针对网络层、协议、主机、网络或端口的过滤，并提供 and、or、not 等逻辑语句来帮助你去掉无用的信息。tcpdump 就是一种免费的网络分析工具，尤其是它提供了源代码，公开了接口，因此具备很强的可扩展性，对于网络维护和入侵者都是非常有用的工具。tcpdump 存在于基本的 Linux 系统中，由于它需要将网络界面设置为混杂模式，普通用户不能正常执行，但具备 root 权限的用户可以直接执行它来获取网络上的信息。因此系统中存在网络分析工具主要不是对本机安全的威胁，而是对网络上的其他计算机的安全存在威胁。另外，由于其相对于 Wireshark 来说，没有非常详细的用户界面，所以非常适合于在终端上使用，网络管理员可以通过常见的命令行来进行流量捕捉和过滤等操作，所以非常方便，这也是它一直广泛被网络管理员欢迎和使用的原因。

### 12.4.2　安装 tcpdump

在 Linux 下 tcpdump 的安装十分简单，一般是以源程序的形式安装。其实，Linux 一个最大的诱人之处就是在其上面有很多软件是提供源程序的，人们可以修改源程序来满足自己的特殊的需要。所以我特别建议读者采取这种源程序的安装方法。最新的 tcpdump 的源代码可以在网站 http://www.tcpdump.org/上即时获得。

在源程序的安装方式中，首先需要取得 tcpdump 的源程序分发包。目前，该源程序包的最新版本为：tcpdump-4.0.0.tar.gz，安装的具体过程如下。

（1）解压缩源代码包：

```
#tar xvfz tcpdump-4.0.0.tar.gz
```

（2）做好编译源程序前的准备活动。

在编译源程序之前，需要确定库文件 Libpcap 已经安装完毕，这个库文件是 tcpdump 软件所需的库文件。同样，你同时还需要一个标准的 C 语言编译器。在 Linux 下标准的 C 语言编译器

一般是 gcc。在 tcpdump 的源程序目录中，有一个文件是 Makefile.in，configure 命令就是从 Makefile.in 文件中自动产生 Makefile 文件。在 Makefile.in 文件中，可以根据系统的配置来修改 BINDEST 和 MANDEST 这两个宏定义，默认值如下。

```
BINDEST = @sbindir@
MANDEST = @mandir@
```

第一个宏值表明安装 tcpdump 的二进制文件的路径名，第二个宏值表明 tcpdump 的 man 帮助页的路径名，用户可以通过修改它们来满足系统的需求。

（3）编译源程序。

使用源程序目录中的 configure 脚本，从系统中读出各种所需的属性。并且根据 Makefile.in 文件自动生成Makefile文件，以便编译使用。make 命令则根据Makefile文件中的规则编译tcpdump 的源程序。使用 make install 命令安装编译好的 tcpdump 的二进制文件。

具体的编译步骤，如下命令所示：

```
# . /configure
# make
# make install
```

### 12.4.3　使用 tcpdump

通常情况下，直接启动 tcpdump 将监视第一个网络界面上所有流过的数据包。如下命令所示：

```
# tcpdump
```

tcpdump 支持相当多的不同参数，如使用-i 参数指定 tcpdump 监听的网络界面，这在计算机具有多个网络界面时非常有用；使用-c 参数指定要监听的数据包数量；使用-w 参数指定将监听到的数据包写入文件中保存，等等。

然而更复杂的 tcpdump 参数是用于过滤目的，这是因为网络中流量很大，如果不加分辨将所有的数据包都截留下来，数据量太大，反而不容易发现需要的数据包。使用这些参数定义的过滤规则可以截留特定的数据包，以缩小目标，才能更好地分析网络中存在的问题。tcpdump 使用参数指定要监视数据包的类型、地址、端口等，根据具体的网络问题，充分利用这些过滤规则就能达到迅速定位故障的目的。请使用 man tcpdump 查看这些过滤规则的具体用法。

#### 1．tcpdump 的选项介绍

tcpdump 的选项非常多，下面给出一些常用的命令选项供读者使用时参考。

- a：将网络地址和广播地址转变成名字。
- -d：将匹配信息包的代码以人们能够理解的汇编格式给出。
- -dd：将匹配信息包的代码以 C 语言程序段的格式给出。
- -ddd：将匹配信息包的代码以十进制的形式给出。
- -e：在输出行打印出数据链路层的头部信息。
- -f：将外部的 Internet 地址以数字的形式打印出来。
- -l：使标准输出变为缓冲行形式。
- -n：不把网络地址转换成名字。

- -t：在输出的每一行不打印时间戳。
- -v：输出一个稍微详细的信息，例如在 IP 包中可以包括 ttl 和服务类型的信息。
- -vv：输出详细的报文信息。
- -c：在收到指定的包的数目后，tcpdump 就会停止。
- -F：从指定的文件中读取表达式，忽略其他的表达式。
- -i：指定监听的网络接口。
- -r：从指定的文件中读取包（这些包一般通过-w 选项产生）。
- -w：直接将包写入文件中，并不分析和打印出来。
- -T：将监听到的包直接解释为指定类型的报文，常见的类型有 RPC（远程过程调用）和 SNMP（简单网络管理协议）。

2．tcpdump 的表达式介绍

表达式是一个正则表达式，tcpdump 利用它作为过滤报文的条件，如果一个报文满足表达式的条件，则这个报文将会被捕获。如果没有给出任何条件，则网络上所有的信息包将会被截获。在表达式中一般有以下几种类型的关键字。

第一种是关于类型的关键字，主要包括 host、net、port，例如 host 211.78.3.2，指明 211.78.3.2 是一台主机，net 210.0.0.0 指明 210.0.0.0 是一个网络地址，port 25 指明端口号是 25。如果没有指定类型，默认的类型是 host。

第二种是确定传输方向的关键字，主要包括 src、dst、dst or src、dst and src，这些关键字指明了传输的方向。举例说明，src 211.78.3.2，指明 IP 包中源地址是 211.78.3.2，dst net 210.0.0.0 指明目的网络地址是 210.0.0.0。如果没有指明方向关键字，则默认是 dst 和 src 关键字。

第三种是协议的关键字，主要包括 fddi、ip、arp、rarp、tcp、udp 等类型。fddi 指明是在 FDDI（分布式光纤数据接口网络）上的特定的网络协议，实际上它是 ether 的别名，fddi 和 ether 具有类似的源地址和目的地址，所以可以将 fddi 协议包当作 ether 的包进行处理和分析。其他的几个关键字就是指明了监听的包的协议内容。如果没有指定任何协议，则 tcpdump 将会监听所有协议的信息包。

除了这三种类型的关键字之外，其他重要的关键字如下：gateway、broadcast、less、greater、还有三种逻辑运算，取非运算是'not '，'! '，与运算是'and'，'&&'，或运算是'or'，'｜｜'。这些关键字可以组合起来构成强大的条件来满足人们的需要。下面举几个例子来进行详细说明。

（1）截获所有 211.78.3.2、主机收到的和发出的所有的数据包：

```
#tcpdump host 211.78.3.2
```

（2）截获主机 211.78.3.2 和主机 211.78.3.3 或 211.78.3.4 的通信，使用命令：

```
#tcpdump host 211.78.3.2 and \ (211.78.3.3 or 211.78.3.4 \)
```

（3）获取除了主机 211.78.3.2 和主机 211.78.3.6 之外所有主机通信的 IP 包，使用如下命令：

```
#tcpdump ip host 211.78.3.2 and ! 211.78.3.6
```

（4）获取主机 211.78.3.1 接收或发出的 telnet 包，使用如下命令：

```
#tcpdump tcp port 23 host 211.78.3.1
```

### 3．查看 tcpdump 的输出结果

由于 tcpdump 的捕包功能强大，因而其输出也是非常丰富的。下面我们介绍几种典型的 tcpdump 命令的输出信息。

（1）查看数据链路层头信息。

使用如下命令：

```
#tcpdump --e host patterson
```

patterson 是一台装有 Linux 的主机，其 MAC 地址是 0:58:46:32:EF:AF，S_Server 是一台装有 SOLARIC 的 SUN 工作站，它的 MAC 地址是 7:43:25:98:6E:AF，上一条命令的输出结果如下：

```
23:49:35.102598 eth0 < 7:43:25:98:6e:af 0:58:46:32:ef:af ip 60: S_Server.33357
> patterson.telnet 0:0(0) ack 22535 win 8760 (DF)
```

分析：23:49:35 是显示的时间，102598 是 ID 号，eth0 <表示从网络接口 eth0 接收该数据包，eth0 >表示从网络接口设备发送数据包，7:43:25:98:6e:af 是主机 S_Server 的 MAC 地址，它表明是从源地址 S_Server 发来的数据包，0:58:46:32:ef:af 是主机 patterson 的 MAC 地址，表示该数据包的目的地址是 patterson.ip，表明该数据包是 IP 数据包，60 是数据包的长度，S_Server.33357>patterson.telnet 表明该数据包是从主机 S_Server 的 33357 端口发往主机 patterson 的 telnet（23）端口，ack 22535 表明对序列号是 22535 的包进行响应，win 8760 表明发送窗口的大小是 8760。

（2）ARP 包的 tcpdump 输出信息。

使用如下命令：

```
#tcpdump arp
```

得到的输出结果是：

```
23:52:42.203783 eth0 > arp who-has route tell patterson (0:58:46:32:ef:af)
23:52:42.204025   eth0   <   arp   reply   route   is-at   0:90:27:12:10:66
(0:58:46:32:ef:af)
```

分析：23:52:42 是时间戳，203783 是 ID 号，eth0 >表明从主机发出该数据包，arp 表明是 ARP 请求包，who-has route tell patterson 表明是主机 patterson 请求主机 route 的 MAC 地址，0:58:46:32:ef:af 是主机 patterson 的 MAC 地址。

（3）TCP 包的输出信息。

用 tcpdump 捕获的 TCP 包的一般输出信息是：

```
src > dst: flags data-seqno ack window urgent options
```

src > dst 表明从源地址到目的地址。flags 是 TCP 包中的标志信息，S 是 SYN 标志，F（FIN），P（PUSH），R（RST），“.”（没有标记）。data-seqno 是数据包中的数据的顺序号。ack 是下次期望的顺序号。window 是接收缓存的窗口大小。urgent 表明数据包中是否有紧急指针。Options 是选项。

（4）UDP 包的输出信息。

用 tcpdump 捕获的 UDP 包的一般输出信息是：

```
route.port1 > patterson.port2: udp length
```

UDP 十分简单，上面的输出行表明从主机 route 的 port1 端口发出的一个 UDP 数据包到主机 patterson 的 port2 端口，类型是 UDP，包的长度是 length。

## 12.5 网络流量分析——NTOP

### 12.5.1 NTOP 介绍

MRTG 基于 SNMP 协议获取信息，对于端口的流量，MRTG 能提供精确统计，但对于三层以上的信息则无从得知了，而这正是 NTOP 的强项。NTOP 能够显示网络的使用情况，它能够显示正在使用网络的主机并报告每个主机发送和接收流量的信息。NTOP 能作为一个前端数据收集器工作（sFlowand/or netFlow），或者作为一个单独的既能收集又能显示的程序工作。查看 NTOP 收集的信息，需要一个浏览器。NTOP 工作在第二层和第三层，默认使用 MAC 地址和 IP 地址。NTOP 能把两者关联起来，这样就能够在网络活动图示里面同时显示 IP 和非 IP 流量（例如：ARP 和 RARP）。NTOP 和 MRTG 相比其安装配置简单。目前市场上可网管型的交换机、路由器都支持 SNMP 协议，NTOP 支持简单网络管理协议，所以可以进行网络流量监控。NTOP 几乎可以监测网络上的所有协议：TCP/UDP/ICMP、(R)ARP、IPX、Telnet、DLC、Decnet、DHCP-BOOTP、AppleTalk、Netbios、TCP/UDP、FTP、HTTP、DNS、SMTP/POP/IMAP、SNMP、NNTP、NFS、X11、SSH 和基于 P2P 技术的协议 eDonkey、Overnet、Bittorrent、Gnutella，Kazaa，等等。

NTOP 工具与 tcpdump 或 Ethereal 工具有极大的差异，它主要是提供网络报文的统计数据，而不是报文的内容。此外，NTOP 不需要使用 Web 服务器，它自身就支持 HTTP 协议。首先它提供了一种快速容易的方法来得到网络活动的准确信息并且不使用网络探测或侦听设备。在大多数情况下，网络探测器对追踪网络故障是必需的。但是在某些情况下，时间是很宝贵的，那么在因为探测器正被使用于监测其他设备而无法获得时，就可以使用 NTOP 工具。其次，在某些给定的网络配置下，不可能与探测器连接，比如一对通过 WAN 互连的 UNIX 系统。在这种情况下，当使用探测器可能很困难时，如果可能，用户应使用 NTOP 工具。

一般来说，NTOP 主要提供以下一些功能。

- 自动从网络中识别有用的信息。
- 将截获的数据包转换成易于识别的格式。
- 对网络环境中通信失败的情况进行分析。
- 探测网络环境中的通信瓶颈。
- 记录网络通信的时间和过程。

NTOP 可以通过分析网络流量来确定网络上存在的各种问题；也可以用来判断是否有黑客正在攻击网络系统；还可以很方便地显示出特定的网络协议、占用大量带宽的主机、各次通信的目标主机、数据包的发送时间、传递数据包的延时等详细信息。通过了解这些信息，网络管理员可以对故障做出及时的响应，对网络进行相应的优化调整，以保证网络运行的效率和安全。

### 12.5.2 安装 NTOP

和 MRTG 相比，NTOP 的安装配置更简单，可以不使用 Apache 服务器。将 NTOP 安装在网

络管理工作站上，监测中、小Linux异构网络的网络性能非常方便。

在安装NTOP之前，首先需要到http://downloads.sourceforge.net/NTOP/NTOP-3.3.8.tar.gz下载NTOP最新的源代码，再到ftp://ftp.rediris.es/sites/ftp.redh...6.2-12.i386.rpm下载相关库函数模块Libpcap。注意：必须先安装Libpcap软件包后才能安装NTOP，具体的安装步骤如下。

（1）安装Libpcap抓包软件包：

```
#rpm -ivh libpcap-0.6.2-12.i386.rpm
```

（2）解压NTOP包：

```
#tar zxvf NTOP-3.3.8.tar.gz
#cd NTOP-3.3.8
```

（3）生成Makefile文件：

```
#./configure
```

（4）配置NTOP时，系统会提示先编译gd和zlib模块。编译完gd和zlib模块，再回到NTOP目录下重新编译、安装：

```
#cd gd-1.8.3/libpng-1.2.1/
#cp scripts/makefile.linux Makefile
#make
#cd ../../zlib-1.1.4
#./configure
#make
#cd ..
#make
#cd NTOP-3.3.8
#make
make install
```

（5）建立NTOP软件需要的log目录存储日志：

```
#mkdir /var/log/NTOP/
```

（6）以上都完成后，就可以启动NTOP了：

```
#NTOP -P /var/log/NTOP/ -u nobody &
```

### 12.5.3 使用NTOP

NTOP支持简单网络管理协议（Simple Network Management Protocol，SNMP），并把PNG格式的图形以HTML的方式显示出来，便于网络管理员对所监控的网络设备（交换机、路由器等）进行管理。

打开浏览器，在地址栏中输入http://host_ip:3000（“IP”就是安装NTOP的那台网络管理工作站的IP地址），即可打开NTOP管理界面。第一次运行时会要求输入管理员的密码，预设密码是“admin”，第二次启动时就不用再输入了。

通过使用NTOP，我们可以在浏览器界面下看到许多有关网络流量的统计和分析信息。下面

介绍几种最重要的统计和分析功能。

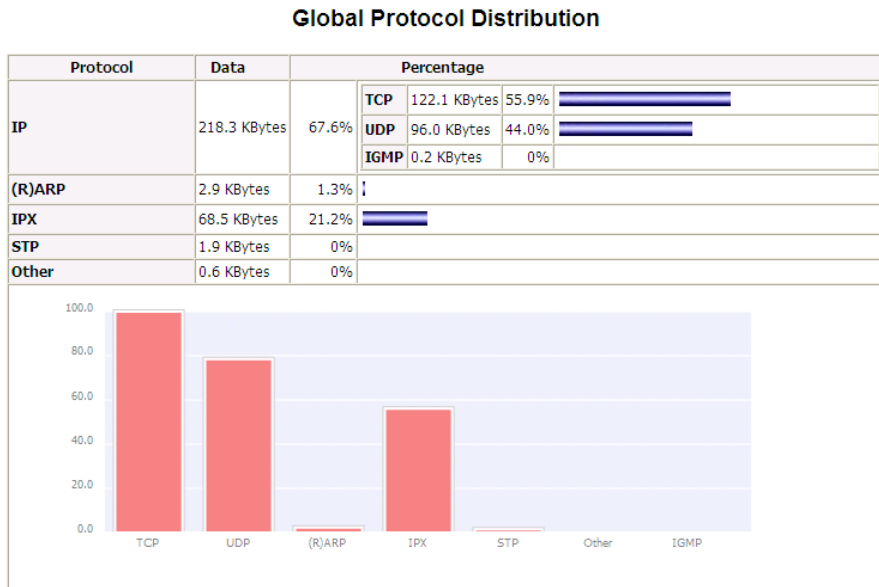
1．查看网络整体流量（Global Traffic）

网络管理员在进行网络流量分析的初始，考虑的大多是宏观的网络整体流量情况，因此，NTOP 首先提供了查看网络整体流量的功能。在 NTOP 的浏览器界面中，查看网络整体流量先切换到 Stats 选项卡，然后单击 Traffic 选项。网络流量会以明细表格的形式显示出来，如图 12-10 所示。另外，图 12-11 和图 12-12 分别向网络管理员显示了网络流量中协议的整体分布情况，以及最常见的传输层协议 TCP 和 UDP 的分布情况，这些可以为网络管理人员的统计、分析和审核工作提供强有力的参考依据。

**Global Traffic Statistics**

| | Name | Device | Type | Speed | Sampling Rate | MTU | Header | Address | IPv6 Addresses |
|---|---|---|---|---|---|---|---|---|---|
| Network Interface(s) | Intel_0 | _Device_NPF_{0CAE5382-ABA2-439E-8234-AB0A6295C99A} | Ethernet | | 0 | 1514 | 14 | 192.168.0.254 | |
| Sampling Since | Wed Oct 29 11:32:53 2008 [2:11] | | | | | | | | |
| Active End Nodes | 43 | | | | | | | | |

图 12-10　Global Traffic Statistics 示意图



**Global Protocol Distribution**

| Protocol | Data | Percentage | | | |
|---|---|---|---|---|---|
| IP | 218.3 KBytes | 67.6% | TCP | 122.1 KBytes | 55.9% |
| | | | UDP | 96.0 KBytes | 44.0% |
| | | | IGMP | 0.2 KBytes | 0% |
| (R)ARP | 2.9 KBytes | 1.3% | | | |
| IPX | 68.5 KBytes | 21.2% | | | |
| STP | 1.9 KBytes | 0% | | | |
| Other | 0.6 KBytes | 0% | | | |



图 12-11　Global Protocol Distribution 示意图

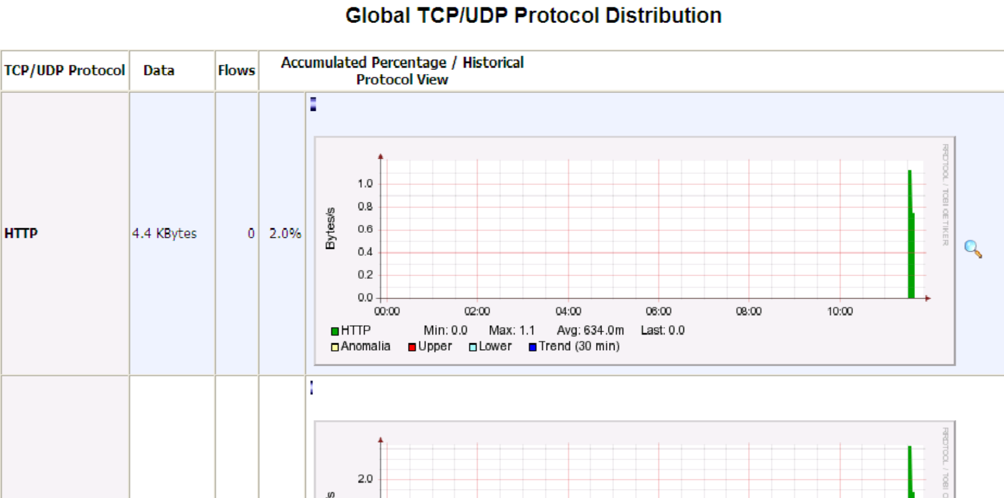
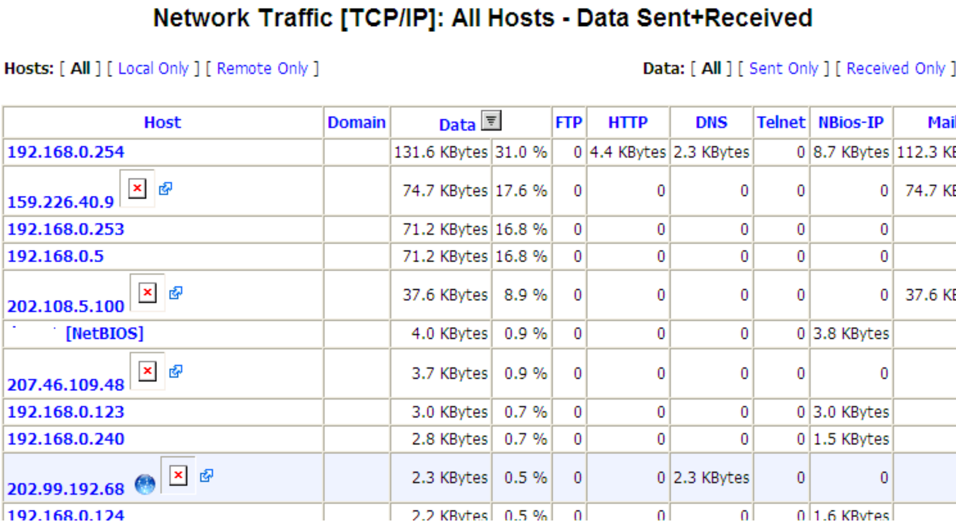



图 12-12　Global TCP/UDP Protocol Distribution 示意图

## 2．查看主机流量

网络管理员在查看了网络整体流量信息的前提下，第二步一般是进一步分析网络中主机的流量情况，确定网络中的用户行为，从而进行安全审计、流量限制等方面的工作。在 NTOP 中，要想查看具体节点计算机的网络流量，切换到 IP Traffic 选项卡，然后单击 Host 选项即可。

- 监测主机使用的网络协议：如图 12-13 所示，网络管理员可以清楚地看到网络中每一台主机针对 FTP、HTTP、DNS 等主要网络协议的使用情况，包括主机的 IP 地址、具体流量等。



**Network Traffic [TCP/IP]: All Hosts - Data Sent+Received**

Hosts: [ All ] [ Local Only ] [ Remote Only ]　　Data: [ All ] [ Sent Only ] [ Received Only ]

| Host | Domain | Data | | FTP | HTTP | DNS | Telnet | NBios-IP | Mail |
|---|---|---|---|---|---|---|---|---|---|
| 192.168.0.254 | | 131.6 KBytes | 31.0 % | 0 | 4.4 KBytes | 2.3 KBytes | 0 | 8.7 KBytes | 112.3 KB |
| 159.226.40.9 | | 74.7 KBytes | 17.6 % | 0 | 0 | 0 | 0 | 0 | 74.7 KB |
| 192.168.0.253 | | 71.2 KBytes | 16.8 % | 0 | 0 | 0 | 0 | 0 | |
| 192.168.0.5 | | 71.2 KBytes | 16.8 % | 0 | 0 | 0 | 0 | 0 | |
| 202.108.5.100 | | 37.6 KBytes | 8.9 % | 0 | 0 | 0 | 0 | 0 | 37.6 KB |
| [NetBIOS] | | 4.0 KBytes | 0.9 % | 0 | 0 | 0 | 0 | 3.8 KBytes | |
| 207.46.109.48 | | 3.7 KBytes | 0.9 % | 0 | 0 | 0 | 0 | 0 | |
| 192.168.0.123 | | 3.0 KBytes | 0.7 % | 0 | 0 | 0 | 0 | 3.0 KBytes | |
| 192.168.0.240 | | 2.8 KBytes | 0.7 % | 0 | 0 | 0 | 0 | 1.5 KBytes | |
| 202.99.192.68 | | 2.3 KBytes | 0.5 % | 0 | 0 | 2.3 KBytes | 0 | 0 | |
| 192.168.0.124 | | 2.2 KBytes | 0.5 % | 0 | 0 | 0 | 0 | 1.6 KBytes | |

图 12-13　Network Traffic 示意图

- 查看网络流量的方向：NTOP 可以把网络主机中的详细出入网络的流量统计和显示出来。这使得网络管理员可以更加清楚地了解网络中主机用户的网络行为，为后续的追踪、审计和流量限制提供了重要信息，如图 12-14 所示。

Remote to Local IP Traffic

| Host | IP Address | Data Sent | | Data Rcvd | |
|---|---|---|---|---|---|
| 7d048c9944a6409 [NetBIOS] | 192.168.0.23 | 3.7 KBytes | 20.0 % | 0 | 0.0 % |
| china-cc6fe71eb [NetBIOS] | 192.168.0.236 | 243 | 1.3 % | 0 | 0.0 % |
| lenovo-fa52b8ff [NetBIOS] | 192.168.0.234 | 252 | 1.3 % | 0 | 0.0 % |
| [NetBIOS] | 192.168.0.171 | 247 | 1.3 % | 0 | 0.0 % |
| [NetBIOS] | 192.168.0.78 | 683 | 3.6 % | 0 | 0.0 % |
| [NetBIOS] | 192.168.0.242 | 243 | 1.3 % | 0 | 0.0 % |
| 192.168.0.37 | 192.168.0.37 | 1.1 KBytes | 5.8 % | 0 | 0.0 % |
| 192.168.0.57 | 192.168.0.57 | 1.1 KBytes | 5.9 % | 0 | 0.0 % |
| 192.168.0.120 | 192.168.0.120 | 92 | 0.0 % | 0 | 0.0 % |
| 192.168.0.123 | 192.168.0.123 | 3.0 KBytes | 16.0 % | 0 | 0.0 % |
| 192.168.0.133 | 192.168.0.133 | 92 | 0.0 % | 0 | 0.0 % |
| 192.168.0.209 | 192.168.0.209 | 7.9 KBytes | 42.6 % | 0 | 0.0 % |

| Total Traffic | Data Sent | Data Rcvd | Used Bandwidth |
|---|---|---|---|
| 18.6 KBytes | 18.6 KBytes | 0 | 1.2 Kbit/s |

图 12-14 Remote to Local IP Traffic 示意图

- 查看网络主机数量和基本统计：NTOP 还具有 Scanner（扫描器）的功能，它能迅速地扫描到网络中的所有活动主机，如图 12-15 所示。

Statistics

| Scanned | |
|---|---|
| Hosts | 60 |
| Less: | |
| No fingerprint | 55 |
| Broadcast | 0 |
| Multicast | 0 |
| Remote | 5 |
| Non IP host | 0 |
| Gives: | |
| Possible to report | 0 |
| Less: Can not resolve* | 0 |
| Less: Unknown Fingerprint** | 0 |

图 12-15 Host Statistics 示意图

总而言之，通过上面的分类讲解我们不难看出，通过使用 NTOP，我们能够对所有进出网络的数据做到了如指掌，并且能够进行非常详细的网络流量分析工作。因此，不管是用来监测网络，还是用来制作网络情况报告，NTOP 都是非常优秀的工具。

# 12.6 网络流量限制——TC 技术

## 12.6.1 TC（Traffic Control）技术原理

Linux 从 kernel 2.1.105 开始支持 QoS（服务质量），不过需要重新编译内核。具体步骤如下。

- 运行 make config 命令时，将 EXPERIMENTAL_OPTIONS 选项设置为 y。
- 将 Class Based Queueing (CBQ)、Token Bucket Flow、Traffic Shapers 选项设置为 y。
- 运行 make dep、make clean、make bzimage，生成新的内核。

在 Linux 操作系统中流量控制器（TC）主要是在输出端口处建立一个队列进行流量控制，控制的方式是基于路由，亦即基于目的 IP 地址或目的子网的网络号的流量控制。流量控制器 TC 其基本的功能模块为队列、分类和过滤器。Linux 内核中支持的队列有 Class Based Queue、Token Bucket Flow、CSZ、First In First Out、Priority、TEQL、SFQ、ATM、RED。由于目前网络流量

种类繁多，网络管理员在管理时通常都采用分类的方式进行，因此，下面所介绍的队列与分类都是基于 CBQ（Class Based Queue）的，而过滤器是基于路由（Route）的，其他的分类方式和过滤器使用方式请参看相关的技术文献。

配置和使用流量控制器 TC，主要分以下几个方面：建立队列、建立分类、建立过滤器和建立路由，另外还需要对现有的队列、分类、过滤器和路由进行监视。

其基本使用步骤如下。

（1）针对网络物理设备绑定一个 CBQ 队列。

（2）在该队列上建立分类。

（3）为每一分类建立一个基于路由的过滤器。

（4）与过滤器相配合，建立特定的路由表。

## 12.6.2　使用 Linux TC 进行流量控制实例

### 1．实例环境及拓扑

在一个局域网中，如图 12-16 所示，我们设定流量控制器上的以太网卡（设备名为 eth0）的 IP 地址为 10.172.4.66，在其上建立一个 CBQ 队列。假设包的平均大小为 1KB，包间隔发送单元的大小为 8B，可接收冲突的发送最长包数目为 20B。

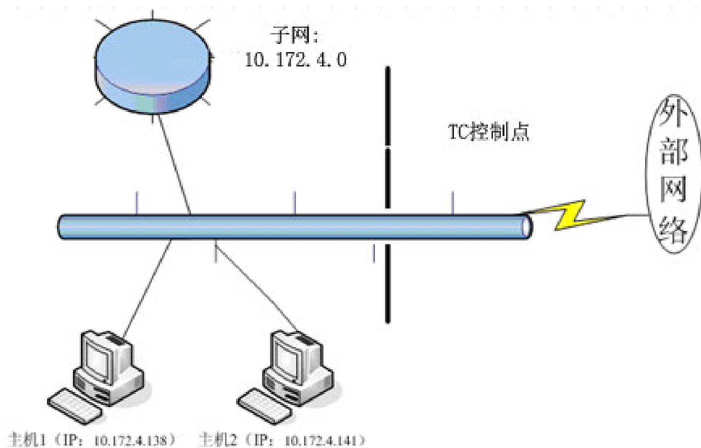



图 12-16　Linux TC 流量控制示意图

假如有以下三种类型的流量需要控制。

（1）发往主机 1 的流量，其 IP 地址设定为 10.172.4.138。其流量带宽控制在 500Mb/s，优先级为 2。

（2）发往主机 2 的流量，其 IP 地址为 10.172.4.141。其流量带宽控制在 200Mb/s，优先级为 1。

（3）发往子网 1 的流量，其子网号为 10.172.4.0，子网掩码为 255.255.255.0。其流量带宽控制在 300Mb/s，优先级为 6。

那么，根据上面的实例条件，我们可以采用以下步骤进行 TC 配置和控制。

### 2．绑定 CBQ 队列

一般情况下，针对一个网卡只需建立一个队列。

将一个 CBG 队列绑定到网络物理设备 eth0 上，其编号为 1:0；网络物理设备 eth0 的实际带宽为 1000Mb/s，包的平均大小为 1000B；包间隔发送单元的大小为 8B，最小传输包大小为 64B。

```
    #tc qdisc add dev eth0 root handle 1: cbq bandwidth 1000Mbit avpkt 1000 cell
8 mpu 64
```

### 3．为队列建立分类

分类建立在队列之上。一般情况下，针对一个队列需建立一个根分类，然后再在其上建立子分类。对于分类，按其分类的编号顺序起作用，编号小的优先；一旦符合某个分类匹配规则，通过该分类发送数据包，则其后的分类不再起作用。

（1）创建根分类 1:1，分配带宽为 1000Mb/s 优先级别为 8。

```
    #tc class add dev eth0 parent 1:0 classid 1:1 cbq bandwidth 1000Mbit rate 1000Mbit
maxburst 20 allot 1514 prio 8 avpkt 1000 cell 8 weight 100Mbit
```

该队列的最大可用带宽为 1000Mb/s，实际分配的带宽为 1000Mb/s，可接收冲突的发送最长包数目为 20B；最大传输单元加 MAC 头的大小为 1514B，优先级别为 8，包的平均大小为 1000B，包间隔发送单元的大小为 8B，相应于实际带宽的加权速率为 100Mbs。

（2）创建分类 1:2，其父分类为 1:1，分配带宽为 500Mb/s，优先级别为 2。

```
    #tc class add dev eth0 parent 1:1 classid 1:2 cbq bandwidth 1000Mbit rate 500Mbit
maxburst 20 allot 1514 prio 2 avpkt 1000 cell 8 weight 50Mbit split 1:0 bounded
```

该队列的最大可用带宽为 1000Mb/s，实际分配的带宽为 500Mb/s，可接收冲突的发送最长包数目为 20 字节；最大传输单元加 MAC 头的大小为 1514B，优先级别为 1，包的平均大小为 1000B，包间隔发送单元的大小为 8B，相应于实际带宽的加权速率为 50Mb/s，分类的分离点为 1:0，且不可借用未使用带宽。

（3）创建分类 1:3，其父分类为 1:1，分配带宽为 200Mb/s，优先级别为 1。

```
    #tc class add dev eth0 parent 1:1 classid 1:3 cbq bandwidth 1000Mbit rate 200Mbit
maxburst 20 allot 1514 prio 1 avpkt 1000 cell 8 weight 20Mbit split 1:0
```

该队列的最大可用带宽为 1000Mb/s，实际分配的带宽为 200Mb/s，可接收冲突的发送最长包数目为 20B；最大传输单元加 MAC 头的大小为 1514B，优先级别为 2，包的平均大小为 1000B，包间隔发送单元的大小为 8B，相应于实际带宽的加权速率为 20Mb/s，分类的分离点为 1:0。

（4）创建分类 1:4，其父分类为 1:1，分配带宽为 300Mb/s，优先级别为 6。

```
    #tc class add dev eth0 parent 1:1 classid 1:4 cbq bandwidth 1000Mbit rate 300Mbit
maxburst 20 allot 1514 prio 6 avpkt 1000 cell 8 weight 30Mbit split 1:0
```

该队列的最大可用带宽为 1000Mb/s，实际分配的带宽为 300Mb/s，可接收冲突的发送最长包数目为 20B；最大传输单元加 MAC 头的大小为 1514B，优先级别为 1，包的平均大小为 1000B，包间隔发送单元的大小为 8B，相应于实际带宽的加权速率为 30Mb/s，分类的分离点为 1:0。

### 4．建立过滤器

过滤器主要服务于分类。一般只需针对根分类提供一个过滤器，然后为每个子分类提供路由映射。

（1）应用路由分类器到 CBQ 队列的根，父分类编号为 1:0；过滤协议为 IP，优先级别为 100，

过滤器为基于路由表。

```
#tc filter add dev eth0 parent 1:0 protocol ip prio 100 route
```

（2）建立路由映射分类 1:2、1:3、1:4。

```
#tc filter add dev eth0 parent 1:0 protocol ip prio 100 route to 2 flowid 1:2
#tc filter add dev eth0 parent 1:0 protocol ip prio 100 route to 3 flowid 1:3
#tc filter add dev eth0 parent 1:0 protocol ip prio 100 route to 4 flowid 1:4
```

### 5．建立与过滤器对应的路由

该路由是与前面所建立的路由映射一一对应的。

（1）发往主机 10.172.4.138 的数据包通过分类 2 转发（分类 2 的速率为 500Mb/s）。

```
#ip route add 10.172.4.138 dev eth0 via 10.172.4.66 realm 2
```

（2）发往主机 10.172.4.30 的数据包通过分类 3 转发（分类 3 的速率为 200Mb/s）。

```
#ip route add 10.172.4.30 dev eth0 via 10.172.4.66 realm 3
```

（3）发往子网 10.172.4.0/24 的数据包通过分类 4 转发（分类 4 的速率为 300Mb/s）。

```
#ip route add 10.172.4.0/24 dev eth0 via 10.172.4.66 realm 4
```

在配置路由时特别值得注意的是：一般对于流量控制器所直接连接的网段建议使用 IP 主机地址流量控制限制，不要使用子网流量控制限制。如果一定需要对直连子网使用子网流量控制限制，则在建立该子网的路由映射前，需将原先由系统建立的路由删除，才可完成相应的步骤。

### 6．对上述建立的机制进行查看

主要包括对现有队列、分类、过滤器和路由的状况进行查看。

（1）显示队列的状况。

简单显示指定设备（这里为 eth0）的队列状况：

```
#tc qdisc ls dev eth0
qdisc cbq 1: rate 1000Mbit (bounded,isolated) prio no-transmit
```

详细显示指定设备（这里为 eth0）的队列状况：

```
#tc -s qdisc ls dev eth0
qdisc cbq 1: rate 1000Mbit (bounded,isolated) prio no-transmit
Sent 7646731 bytes 13232 pkts (dropped 0, overlimits 0)
borrowed 0 overactions 0 avgidle 31 undertime 0
```

这里主要显示了通过该队列发送了 13232 个数据包，数据流量为 7646731B，丢弃的包数目为 0，超过速率限制的包数目为 0。

（2）显示分类的状况。

简单显示指定设备（这里为 eth0）的分类状况：

```
#tc class ls dev eth0
class cbq 1: root rate 1000Mbit (bounded,isolated) prio no-transmit
class cbq 1:1 parent 1: rate 10Mbit prio no-transmit #no-transmit 表示优先级为 8
```

```
class cbq 1:2 parent 1:1 rate 500Mbit prio 2
class cbq 1:3 parent 1:1 rate 200Mbit prio 1
class cbq 1:4 parent 1:1 rate 300Mbit prio 6
```

详细显示指定设备（这里为 eth0）的分类状况：

```
#tc -s class ls dev eth0
class cbq 1: root rate 1000Mbit (bounded,isolated) prio no-transmit
Sent 17725304 bytes 32088 pkts (dropped 0, overlimits 0)
borrowed 0 overactions 0 avgidle 31 undertime 0
class cbq 1:1 parent 1: rate 1000Mbit prio no-transmit
Sent 16627774 bytes 28884 pkts (dropped 0, overlimits 0)
borrowed 16163 overactions 0 avgidle 587 undertime 0
class cbq 1:2 parent 1:1 rate 500Mbit prio 2
Sent 628829 bytes 3130 pkts (dropped 0, overlimits 0)
borrowed 0 overactions 0 avgidle 4137 undertime 0
class cbq 1:3 parent 1:1 rate 200Mbit prio 1
Sent 0 bytes 0 pkts (dropped 0, overlimits 0)
borrowed 0 overactions 0 avgidle 159654 undertime 0
class cbq 1:4 parent 1:1 rate 300Mbit prio 6
Sent 5934679 bytes 9354 pkts (dropped 0, overlimits 0)
borrowed 3797 overactions 0 avgidle 159557 undertime 0
```

这里主要显示了通过不同分类发送的数据包、数据流量、丢弃的包数目、超过速率限制的包数目，等等。其中根分类（class cbq 1:0）的状况应与队列的状况类似。

例如，分类 class cbq 1:4 发送了 9354 个数据包、数据流量为 5934679B、丢弃的包数目为 0、超过速率限制的包数目为 0。

（3）显示过滤器的状况：

```
#tc -s filter ls dev eth0
filter parent 1: protocol ip pref 100 route
filter parent 1: protocol ip pref 100 route fh 0xffff0002 flowid 1:2 to 2
filter parent 1: protocol ip pref 100 route fh 0xffff0003 flowid 1:3 to 3
filter parent 1: protocol ip pref 100 route fh 0xffff0004 flowid 1:4 to 4
```

这里 flowid 1:2 代表分类 class cbq 1:2，to 2 代表通过路由 2 发送。

（4）显示现有路由的状况：

```
#ip route
10.172.4.66 dev eth0 scope link
10.172.4.138 via 10.172.4.66 dev eth0 realm 2
10.172.4.30 via 10.172.4.66 dev eth0 realm 3
```

```
10.172.4.0/24 via 10.172.4.66 dev eth0 realm 4
10.172.4.0/24 dev eth0 proto kernel scope link src 10.172.4.66
172.16.1.0/24 via 10.172.4.66 dev eth0 scope link
127.0.0.0/8 dev lo scope link
default via 10.172.4.254 dev eth0
```

7．队列、分类、过滤器及路由维护

上面我们通过一个完整的例子示意了使用 Linux 的 TC 进行流量控制的全过程。不难看出，该技术主要包括如上 5 步。而在日常的网络管理过程中，网管员还需要对 TC 的队列、分类、过滤器和路由进行相应的增添、修改和删除等操作，以保证流量控制能够因时、因地、因应用制宜。可能用到的相应命令如下。

- tc class add 命令：添加分类。
- tc class change 命令：修改分类。
- tc filter add 命令：添加过滤器。
- tc filter change 命令：修改过滤器。
- tc filter del 命令：删除过滤器。
- ip route add 命令：添加与过滤器对应的路由。
- ip route change 命令：修改与过滤器对应的路由。
- ip route del 命令：删除与过滤器对应的路由。

具体的用法和例子在此处不再一一给出，请参看详细的技术文献进行对照使用。

## 12.7 网络流量管理的策略

### 12.7.1 网络流量管理的目标

随着网络流量的不断增长以及网络应用的日趋纷繁复杂化，我们不难看到，简单地无限制增加网络带宽是不能解决网络流量的根本问题的。我们需要对网络流量进行管理，从而保证网络的健康和网络应用的正常服务。在网络流量管理的过程中，我们首要的问题就是要明确网络管理的目标。在网络流量管理中，我们需要牢记以下四个目标：

- 要了解网络流量的使用情况。
- 要找到优化网络性能的途径。
- 要通过网络管理技术来提升网络效能。
- 需要做好网络流量信息安全方面的防护工作。

具体来说，要达到上述四个目标，网络管理员首先可以通过有效的分类方式非常明确地知道我们需要的带宽到底哪些是实际使用的。网络流量管理的第二个目标是找到网络性能的瓶颈。网络性能很重要的一个方面是吞吐量，这是网络能够传输的最大数据量，还有延迟等。第三，通过本章所介绍的流量监控及控制软件可以高效地提升网络性能，从而满足不同的网络应用需求。最后，网络管理员还可以综合运用入侵检测系统（IDS）、防火墙（FireWall）、统一威胁管理（UTM）

设备来对网络流量进行信息安全方面的防护工作。当然，信息安全方面的工作不是本章介绍的范畴，在这里不作过多介绍。

### 12.7.2 网络流量管理的具体策略

在日常的网络流量管理中，为了有效实现网络管理的四个目标，我们需要采取相应的步骤。这个步骤分别是：网络流量捕捉和分类、网络流量监视（统计和分析）和控制策略。

- 网络流量捕捉和分类。这是进行网络流量管理的第一步。只有通过设置捕捉点，对网络流量进行捕捉和分类，才能进行后续的分析和控制工作。这里特别需要强调的是，网络流量分类可以宏观化，也可以细化。比如 TCP、UDP、ICMP 等的分类就比较宏观，而 HTTP、FTP 甚至是诸如 Kazza、Skype 等 P2P 流量的分类和识别就比较细化了。本章介绍的 Wireshark 和 tcpdump 软件目前着重的是宏观流量的捕捉和分类，具体细化的内容管理员可以参看相关的资料获得。
- 网络流量监视（分析）。监视步骤用来显示流量的运行状况，帮助找出问题所在和执行相应的管理策略。应用程序和网络管理能够收集分类、展示和收集信息，包括带宽利用率、活跃的主机和网络效率以及活跃的应用程序。我们的设备能够跟踪平均和高信息的流量，识别最大的用户和应用程序，将网络流量定位到不同的领域，从应用的角度监视网络流量，分析网络带宽明确的关键问题所在。用统计报表来进行表现。该目标可以采用本章所介绍的 NTOP 可视化分析管理工具来协助网络管理员在实际工作中实现。
- 控制策略。通过网络流量分析后，接下来根据优先级别分配带宽资源。分配的依据可以根据主机、应用，等等，特别需要考虑的是将消耗资源的 P2P 程序或者音视频下载等进行滞后考虑。用户可以根据本章所介绍的 TC 工具来进行和实现一个完整的分类监视和控制网络流量，这样，我们就可以将网络流量有效管理起来，将原来无序的网络流量变得有序起来。

# 第 13 章

# 兵来将挡，水来土掩：企业级防火墙部署及应用

本章导读

防火墙一直是信息安全领域的标志性产品，它可以在网络层甚至是目前的应用层来对恶意流量进行封堵，从而保证企业网及其服务的安全。在 Linux 领域，Netfilter/Iptables 防火墙框架比较成熟，500 强企业完全可以采用该框架来进行企业防护。

当然，防火墙的部署和使用具有一定的技巧，比如 DMZ 的部署、Iptables 封堵规则的设定等，都需要审慎对待才能发挥其最大功效。基于此，本章将详细介绍企业级防火墙的部署及应用。

## 13.1 防火墙技术简介

请参考 2.3.1 节。

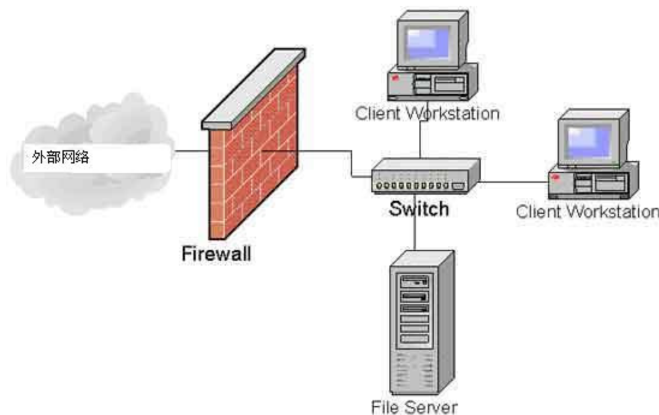



图 13-1　防火墙体系网络结构

## 13.2 Netfilter/Iptables 防火墙框架技术原理

Linux 系统提供了一个自带免费的防火墙——Netfilter/Iptables 防火墙框架，该框架功能强大。下面详细介绍该框架的安装、配置和使用。

### 13.2.1　Linux 中的主要防火墙机制演进

Linux 的防火墙技术经历了若干代的改革，一步步地发展而来。最开始的 Ipfwadm 是 Alan Cox 在 Linux kernel 发展的初期，从 FreeBSD 的内核代码中移植过来的。后来经历了 Ipchains，再经由 Paul Russell 在 Linux kernel 2.3 系列的开发过程中发展了 Netfilter 这个架构。而用户空间的防火墙管理工具，也相应的发展为 Iptables。Netfilter/Iptables 这个组合目前相当令人满意。在经历了 Linux kernel 2.4 和 2.5 的发展以后，的确可以说，Netfilter/Iptables 经受住了大量用户广泛使用的考验。准确地说，Netfilter/Iptables 可以说是 Linux 的第三代防火墙，而且它比 Ipfwadm 和 Ipchains 要出色很多，因此，Linux 就将 Netfilter 设定为其预定的防火墙，虽然 Netfilter 是自由软件，但其功能、稳定性、安全性、扩展性等方面都丝毫不逊于其他的商业防火墙。

### 13.2.2　Netfilter/Iptables 架构简介

Netfilter/Iptables 可以对流入和流出的信息进行细化控制，且可以在一台低配置机器上很好地运行，被认为是 Linux 中实现包过滤功能的第四代应用程序。Netfilter/Iptables 包含在 Linux 2.4 以后的内核中，可以实现防火墙、NAT（网络地址翻译）和数据包的分割等功能。Netfilter 工作在内核内部，而 Iptables 则是让用户定义规则集的表结构。Netfilter/Iptables 从 Ipchains 和 Ipwadfm（IP 防火墙管理）演化而来，功能更加强大。

这里所说的 Iptables 是 Ipchains 的后继工具，但具有更强的可扩展性。内核模块可以注册一个新的规则表（table），并要求数据包流经指定的规则表。这种数据包选择用于实现数据包过滤（filter 表）、网络地址转换（NAT 表）及数据包处理（mangle 表）。Linux 2.4 内核及其以上版本提供的这三种数据包处理功能都基于 Netfilter 的钩子函数和 IP 表，都是相互间独立的模块，完美地集成到了由 Netfilter 提供的框架中，如图 13-2 所示。Netfilter 主要提供了以下三项功能。

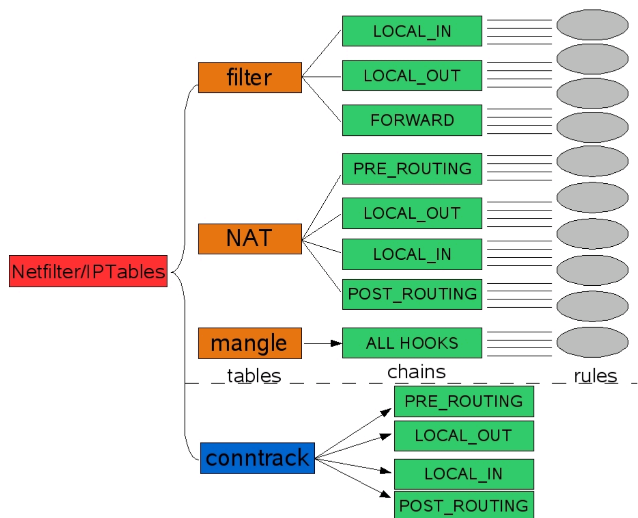



图 13-2　Netfilter/Iptables 框架结构示意图

- 包过滤。filter 表格不会对数据包进行修改，而只对数据包进行过滤。Iptables 优于 Ipchains 的一个方面就是它更为小巧和快速。它是通过钩子函数 NF_IP_LOCAL_IN、NF_IP_FORWARD 及 NF_IP_LOCAL_OUT 接入 Netfilter 框架的。
- NAT。NAT 表格监听三个 Netfilter 钩子函数：NF_IP_PRE_ROUTING、NF_IP_POST_ROUTING 及 NF_IP_LOCAL_OUT。NF_IP_PRE_ROUTING 实现对需要转发数据包的源地址进行地址转换，而 NF_IP_POST_ROUTING 则对需要转发的数据包的目的地址进行地址转换。对于本地数据包目的地址的转换，则由 NF_IP_LOCAL_OUT 来实现。
- 数据包处理。mangle 表格在 NF_IP_PRE_ROUTING 和 NF_IP_LOCAL_OUT 钩子中进行注册。使用 mangle 表，可以实现对数据包的修改或给数据包附上一些外带数据。当前 mangle 表支持修改 TOS 位及设置 skb 的 nfmard 字段。

根据实际情况，灵活运用 Netfilter/Iptables 框架，生成相应的防火墙规则可以方便、高效地阻断部分网络攻击以及非法数据包，参见图 13-3 所示的工作原理。然而，由于配置了防火墙，可能会引起诸如 FTP、QQ、MSN 等协议和软件无法使用或者某些功能无法正常使用，也有可能引起 RPC（远程过程调用）无法执行，这需要用户根据实际情况来配置相应的服务代理程序来开启这些服务。需要特别提醒注意的是，防火墙也可能被内部攻击，其并不是万能的，还需要综合使用其他防护手段。内部人员由于无法通过 Telnet 浏览邮件或

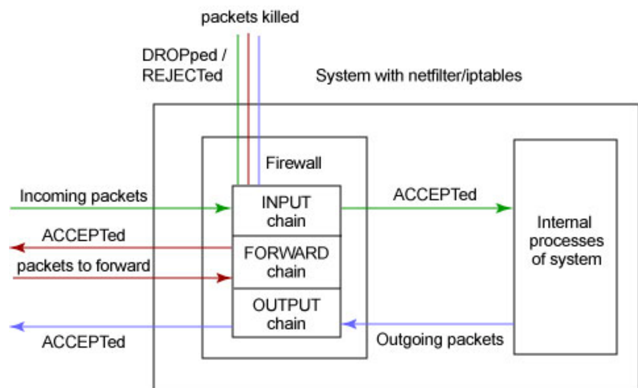



图 13-3　Netfilter/Iptables 框架工作原理

使用 FTP 向外发送信息，个别人会对防火墙不满进而可能对其进行攻击和破坏。而且，攻击的目标常常是防火墙或防火墙运行的操作系统，这极大地危害了防火墙系统甚至是关键信息系统的安全。

### 13.2.3 Netfilter/Iptables 模块化工作架构

Netfilter 是工作在 Linux 中的一个功能，而这种功能是通过“模块”化的方式来扩充和实现的。用户可以通过 Linux 的模块管理工具随心所欲地将模块载入内存，或者将某个不需要使用的模块从内存中删除掉，而 Netfilter 也是以模块的方式存在于 Linux 之中的。可以认为，Linux 系统内存中多加载了一个 Netfilter 模块就意味着 Linux 防火墙的功能多了一项。

Netfilter 在 Linux 系统中有两种存在方式：与 IPv4 或者 IPv6 协议相关；与 IPv4 或者 IPv6 协议均无关。如下所示的目录结构反映了与 IPv4 协议相关的 Netfilter 模块的存在方式：

```
    /lib/modules/2.6.18-8.el5/kernel/net/ipv4/netfilter
    arptable_filter.ko ip_conntrack_proto_sctp.ko  ip_nat_tftp.ko ipt_DSCP.ko
ipt_REJECT.ko
    arp_tables.ko ip_conntrack_sip.ko ip_queue.ko ipt_ecn.ko ipt_SAME.ko
    arpt_mangle.ko ip_conntrack_tftp.ko iptable_filter.ko ipt_ECN.ko ipt
_TCPMSS.ko
    ip_conntrack_amanda.ko ip_nat_amanda.ko iptable_mangle.ko ipt_hashlimit.ko
ipt_tos.ko
    ip_conntrack_ftp.ko ip_nat_ftp.ko iptable_nat.ko ipt_iprange.ko ipt_TOS.ko
    ip_conntrack_h323.koip_nat_h323.ko iptable_raw.koipt_LOG.ko ipt_ttl.ko
    ip_conntrack_irc.ko ip_nat_irc.ko ip_tables.ko ipt_MASQUERADE.ko  ipt_TTL.ko
    ip_conntrack.ko ip_nat.ko ipt_addrtype.ko ipt_NETMAP.ko ipt_ULOG.ko
    ip_conntrack_netbios_ns.ko ip_nat_pptp.ko ipt_ah.ko ipt_owner.ko
    ip_conntrack_netlink.ko ip_nat_sip.ko ipt_CLUSTERIP.ko ipt_recent.ko
    ip_conntrack_pptp.ko ip_nat_snmp_basic.ko ipt_dscp.ko ipt_REDIRECT.ko
```

如下所示的目录结构反映了与 IPv6 协议相关的 Netfilter 模块的存在方式：

```
    /lib/modules/2.6.18-8.el5/kernel/net/ipv6/netfilter
    ip6_queue.ko ip6table_raw.ko ip6t_dst.ko ip6t_hbh.ko ip6t_ipv6header.ko
ip6t_REJECT.ko
    ip6table_filter.ko ip6_tables.ko ip6t_eui64.ko ip6t_hl.ko ip6t_LOG.ko
ip6t_rt.ko
    ip6table_mangle.ko ip6t_ah.ko ip6t_frag.ko ip6t_HL.ko ip6t_owner.ko
```

以上两个目录结构中的模块均只能在 IPv4 和 IPv6 协议下工作，它们是 Linux 内核版本 2.6.14 下的产物。而为了使得对 Netfilter 模块的管理更加简单和高效，从 Linux 2.6.14 内核版本之后，Netfilter 的模块就只有如下的目录结构方式，它们都是与协议无关的，因而能够工作在 IPv4 和 IPv6 网络协议环境下：

```
/lib/modules/2.6.18-8.el5/kernel/net/netfilter
    nfnetlink.ko xt_connbytes.ko xt_esp.ko xt_MARK.ko xt_policy.ko xt
_statistic.ko
    nfnetlink_log.ko xt_connmark.ko xt_helper.ko xt_multiport.ko xt_quota.ko
xt_string.ko
    nfnetlink_queue.ko xt_CONNMARK.ko xt_length.ko xt_NFQUEUE.ko xt_realm.ko
xt_tcpmss.ko
    x_tables.ko xt_CONNSECMARK.ko xt_limit.ko xt_NOTRACK.ko xt_sctp.ko
xt_tcpudp.ko
    xt_CLASSIFY.ko xt_conntrack.ko xt_mac.ko xt_physdev.ko xt_SECMARK.ko
    xt_comment.ko xt_dccp.ko xt_mark.ko xt_pkttype.ko xt_state.ko
```

### 13.2.4 安装和启动 Netfilter/Iptables 系统

因为 Netfilter/Iptables 的 Netfilter 组件是与内核 2.4.x 集成在一起的，对于 Red Hat Linux 9 或更高版本的 Linux 都配备了 Netfilter 这个内核工具，所以一般不需要下载，而只要下载并安装 Iptables 用户空间工具的源代码包，下载的网址为：http://www.netfilter.org/projects/iptables/downloads.html。目前，最新源代码安装包是：iptables-1.4.2-rc1.tar.bz2。在 Red Hat Enterprise Linux 5 中，已经自带了 Iptables 用户空间工具，不需要自己下载源代码安装，这里只是对源代码安装作一个介绍。

在开始安装 Iptables 用户空间工具之前，要对系统做某些修改。主要有以下选项需要进行配置和修改。

- CONFIG_PACKET：如果要使应用程序直接使用某些网络设备，那么该选项是有用的。
- CONFIG_IP_NF_MATCH_STATE：如果要配置有状态的防火墙，那么该选项非常重要而且很有用。这类防火墙会记得先前关于信息包过滤所做的决定，并根据它们做出新的决定。
- CONFIG_IP_NF_FILTER：该选项提供一个基本的信息包过滤框架。如果打开该选项，则会将一个基本过滤表（带有内置的 INPUT、FORWARD 和 OUTPUT 链）添加到内核空间。
- CONFIG_IP_NF_TARGET_REJECT：该选项允许指定应该发送 ICMP 错误消息来响应已被 DROP 掉的入站信息包，而不是简单地杀死这些信息包。

下面是安装源代码包的步骤：

```
//将源代码文件解压缩
#bzip2 -d iptables-1.4.2-rc1.tar.bz2
#tar -xvf iptables 1.4.2.tar
//切换目录
#cd iptables 1.4.2
//编译该工具，指定编译的内核目录为/usr/src/linux-2.6.18-8
```

```
#make KERNEL_DIR=/usr/src/linux-2.6.18-8
//执行 make install 命令，同样设定内核目录为/usr/src/linux-2.6.18-8
#make install KERNEL_DIR=/usr/src/linux-2.6.18-8
```

安装完成后，就可以启动防火墙了。下面启动 Iptables 命令：

```
//使用 service 命令手工启动
# service iptables start
```

如果想要在系统启动的时候也启动该防火墙服务，那么可以使用 setup 命令，然后进入 System service 选项，选择 iptables 守护进程即可。

### 13.2.5 使用 Iptables 编写防火墙规则

通过向防火墙提供有关对来自某个源、到某个目的地或具有特定协议类型的信息包要做些什么的指令，规则控制信息包的过滤。通过使用 Netfilter/Iptables 系统提供的特殊命令 Iptables，建立这些规则，并将其添加到内核空间的特定信息包过滤表内的链中。关于添加/除去/编辑规则的命令的一般语法如下：

```
iptables [-t table] command [match] [target]
```

不难看出，一条 Iptables 规则包含以下 5 个基本元素。

- 表
- 命令
- 链
- 匹配
- 动作

#### 1. 表（table）

-t table 选项允许使用标准表之外的任何表。表是包含仅处理特定类型信息包的规则和链的信息包过滤表。有三种可用的表选项：filter、nat 和 mangle。该选项不是必选的，如果未指定，则 filter 用作默认表。filter 表用于一般的信息包过滤，包含 INPUT、OUTPUT 和 FORWAR 链。nat 表用于要转发的信息包，它包含 PREROUTING、OUTPUT 和 POSTROUTING 链。如果信息包及其头内进行了任何更改，则使用 mangle 表。该表包含一些规则来标记用于高级路由的信息包以及 PREROUTING 和 OUTPUT 链。

#### 2. 命令（command）

command 部分是 Iptables 命令的最重要部分，它告诉 Iptables 命令要做什么，例如，插入规则、将规则添加到链的末尾或删除规则。主要有如表 13-1 所示的命令。

表 13-1　Iptables 常用命令

| 命　令 | 说　明 |
| --- | --- |
| -A 或--append | 该命令将一条规则附加到链的末尾 |
| -D 或--delete | 通过用-D 指定要匹配的规则或者指定规则在链中的位置编号，该命令从链中删除该规则 |

续表

| 命　令 | 说　明 |
| --- | --- |
| -P 或--policy | 该命令设置链的默认目标，即策略。所有与链中任何规则都不匹配的信息包都将被强制使用此链的策略 |
| -N 或--new-chain | 用命令中所指定的名称创建一个新链 |
| -F 或--flush | 如果指定链名，该命令删除链中的所有规则，如果未指定链名，该命令删除所有链中的所有规则。此参数用于快速清除 |
| -L 或--list | 列出指定链中的所有规则 |
| -R 或--replace | 替换指定链中一条匹配的规则 |
| -X 或--delete-chain | 删除指定用户的定义链，若没有指定链，则删除所有的用户链 |
| -C 或--check | 检查数据包是否与指定链的规则相匹配 |
| -Z 或--zero | 将指定链中所有规则的 byte 计数器清零 |

3．匹配（match）

Iptables 命令的可选 match 部分指定信息包与规则匹配所应具有的特征（如源和目的地地址、协议等）。匹配分为两大类：通用匹配和特定于协议的匹配。这里，将研究可用于采用任何协议的信息包的通用匹配。下面是一些重要的且常用的通用匹配及其说明，如表 13-2 所示。

表 13-2　通用匹配说明

| 通用匹配 | 说　明 |
| --- | --- |
| -p 或--protocol | 该通用协议匹配用于检查某些特定协议。协议示例有 TCP、UDP、ICMP，用逗号分隔的任何这三种协议的组合列表以及 ALL（用于所有协议）。ALL 是默认匹配。可以使用“!”符号表示不与该项匹配 |
| -s 或 --source | 该源匹配用于根据信息包的源 IP 地址来与它们匹配。该匹配还允许对某一范围内的 IP 地址进行匹配，可以使用“!”符号，表示不与该项匹配。默认源匹配与所有 IP 地址匹配 |
| -d 或 --destination | 该目的地匹配用于根据信息包的目的地 IP 地址来与它们匹配。该匹配还允许对某一范围内 IP 地址进行匹配，可以使用“!”符号表示不与该项匹配 |
| --sport | 指定匹配规则的源端口或端口范围 |
| --dport | 指定匹配规则的目的端口或端口范围 |
| -i | 匹配单独的网络接口或某种类型的接口设置过滤规则 |

4．目标（target）

前面已经讲过，目标是由规则指定的操作，对与那些规则匹配的信息包执行这些操作。除了允许用户定义的目标之外，还有许多可用的目标选项。下面是常用的一些目标及其示例和说明，如表 13-3 所示。

表 13-3　目标项说明

| 目 标 项 | 说　明 |
| --- | --- |
| ACCEPT | 当信息包与具有 ACCEPT 目标的规则完全匹配时，会被接收（允许它前往目的地） |
| DROP | 当信息包与具有 DROP 目标的规则完全匹配时，会阻塞该信息包，并且不对它做进一步处理。该目标被指定为-j DROP |
| REJECT | 该目标的工作方式与 DROP 目标相同，但它比 DROP 好。和 DROP 不同，REJECT 不会在服务器和客户机上留下死套接字。另外，REJECT 将错误消息发回给信息包的发送方。该目标被指定为-j REJECT |
| RETURN | 在规则中设置的 RETURN 目标让与该规则匹配的信息包停止遍历包含该规则的链。如果链是如 INPUT 之类的主链，则使用该链的默认策略处理信息包。它被指定为-jump RETURN |
| LOG | 表示将包的有关信息记录入日志 |
| TOS | 表示改写数据包的 TOS 值 |

## 13.3 使用 Iptables 编写规则的简单应用

### 1．编写规则示例一：基本规则

下面将给出运用上述框架理论形成规则的一些简单示例，以供读者在实际的应用过程中进行模仿和使用。

（1）接收来自指定 IP 地址的所有流入的数据包：

```
#iptables -A INPUT -s 203.159.0.10 -j ACCEPT
```

（2）只接收来自指定端口（服务）的数据包：

```
#iptables -D INPUT --dport 80 -j DROP
```

（3）允许转发所有到本地（198.168.10.13）SMTP 服务器的数据包：

```
#iptables -A FORWARD -p tcp -d 198.168.10.13 --dport smtp -i eth0 -j ACCEPT
```

（4）允许转发所有到本地的 UDP 数据包（诸如即时通信等软件产生的数据包）：

```
#iptables -A FORWARD -p udp -d 198.168.80.0/24 -i eth0 -j ACCEPT
```

（5）拒绝发往 WWW 服务器的客户端的请求数据包：

```
#iptables -A FORWARD -p tcp -d 198.168.80.11 --dport www -i eth0 -j REJECT
```

（6）允许目的地为指定端口的 TCP 数据包进入：

```
#iptables -A INPUT -p tcp -m multiport --destination-port 21,53,80,25,110
ACCEPT
```

（7）允许来源为指定端口的 TCP 数据包进入：

```
#iptables -A INPUT -p tcp -m multiport --source-port 21,53,80,25,110 ACCEPT
```

（8）丢掉 SYN 和 ACK 标志位置位的数据包：

```
#iptables -A INPUT-p tcp --tcp-flags ALL SYN,ACK DROP
```

### 2．编写规则示例二：进行碎片检测及流量控制

（1）检查 IP 碎片。在 TCP/IP 网络中，链路层具有最大传输单元 MTU 这个特性，它限制了数据帧的最大长度，不同的网络类型都有一个上限值。以太网的 MTU 是 1500。如果 IP 层有数据包要传送，而且数据包的长度超过了 MTU，那么 IP 层就要对数据包进行分片（fragmentation）操作，使每一片的长度都小于或等于 MTU，这些被分段的片段就成为 IP 碎片。那么，如果在防火墙处不对 IP 碎片进行特别处理的话，那么有可能部分 IP 碎片会被防火墙拦截，从而影响到接收端对这些碎片的还原，并最终影响到信息的完整性和可用性问题，所以，下面的例子给出防火墙允许 IP 碎片通过的规则：

```
#iptables -A FORWARD -p tcp -f -s 172.168.96.0/24 -d 172.168.97.18 -j ACCEPT
```

需要特别留意上述规则中的-f 选项，它指定了第二个以及以后的 IP 碎片将由防火墙来处理通过，否则的话，考虑下面的规则，防火墙有可能对其第二个及其以后的 IP 碎片进行拦截，从而影响正常的信息流通：

```
#iptables -A FORWARD -p tcp -s 172.168.96.0/24 -d 172.168.97.18 -j ACCEPT
```

（2）速率限制。Iptables 提供了非常健全的速率控制机制，主要用来限制由外向内的单位时间内通过的数据包个数，这样做的一个直接好处就是尽可能地抑制前面多次提到的拒绝服务攻击或者是分布式拒绝服务攻击，因为这两种攻击的一个非常典型的表现就是单位时间内有很多数据包涌向目的地。所以，我们可以使用下面的规则来限制单位时间内允许通过防火墙，从而进入被保护网络的数据包个数：

```
#iptables -A INPUT -m limit --limit 200/second
#iptables -A INPUT -m limit --limit 10000/minute
```

上述两条规则分别限制 1 秒内和 1 分钟内通过的数据包个数不能超过 200 和 10000 个。当然，在实际应用中，也可以通过/second、/minute、/hour、/day 这样的时间间隔来进行设定，并且，其中诸如 200 和 10000 这些具体数值的设定需要用户根据具体情况和经验来进行设定，没有规定的数值可循。

另外，在设定速率限制后，还可以设定超过该限制所触发的一些处理事件，比如说直接丢弃。下面的规则表示当速率超过 200 限制后，将直接对后续数据包进行丢弃：

```
#iptables -A INPUT -m limit --limit-burst 200
```

### 3．编写规则示例三：保障网络服务安全

在上一小节我们根据 Iptables 的规则编写原则介绍了一些简单的使用其进行数据包过滤的例子，由于 Iptables 在实际生活中得到了非常广泛的使用，本节将介绍一个使用其保障网络服务安全的例子，以使用户能够在实际使用中举一反三。

1）具体应用场景

在这个应用中，我们将需要使用 Iptables 防火墙保护企业网络提供的对外的公共 Internet 服务，这些服务包括 WWW 服务、FTP 服务、SMTP 服务以及 DNS 服务（具体的网络拓扑请参见图 13-4）。因此，这些服务都具有有效的 Internet 地址。

为了将内部网段 210.10.18.0/24 与 Internet 隔离，在内部网络和 Internet 之间使用了包过滤防火墙。具体的 IP 地址设置如下。

- 防火墙的内网接口是 eth0（IP 地址为：210.10.18.88），防火墙对外的 Internet 接口是 eth1（IP 地址为：210.10.19.188）。
- WWW 服务器：IP 地址为 210.10.18.89。
- FTP 服务器：IP 地址为 210.10.18.90。
- DNS 服务器：IP 地址为 210.10.18.91。
- SMTP 服务器：IP 地址为 210.10.18.92。

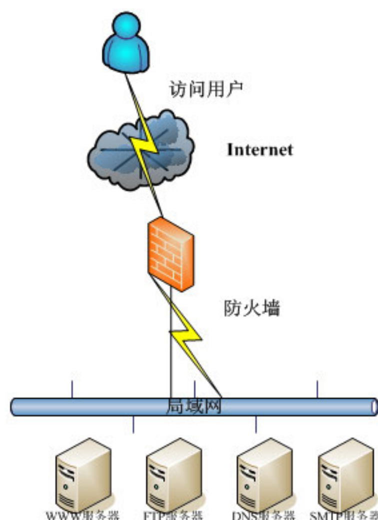


图 13-4　防火墙部署实例示意图

2）防火墙具体配置

本应用的主要目的是对企业网提供各种服务的服务器提供保护，以使它们免受来自于外网 Internet 恶意用户和流量的攻击和危害。在配置过程中，需要根据本章前面部分介绍的知识进行规则编写等工作具体的步骤如下。

（1）建立有关的脚本文件：在/etc/rc.d/目录下用 touch 命令建立空的脚本文件，执行 chmod 命令添加可执行权限。

```
# touch /etc/rc.d/firewall-for-networkservice
# chmod u+x /etc/rc.d/firewall-for-networkservice
```

（2）编辑/etc/rc.d/rc.local 文件，在末尾加上/etc/rc.d/firewall-for-networkservice 以确保开机时能自动执行该脚本，运行如下命令：

```
# echo "/etc/rc.d/firewall-for-networkservice" >>/etc/rc.d/rc.local
```

（3）使用 Vi 或者 Gedit 等编辑器编辑/etc/rc.d/firewall-for-networkservice 文件，插入如下内容。

① 添加相关脚本信息。注意：脚本中的注释是采用“#”表示，而不是通常用的“//”。

```
# 添加脚本编写头部
# !/bin/bash
# 在屏幕上显示信息
echo "Starting iptables rules…"
# 开启内核转发功能
echo "1" >/proc/sys/net/ipv4/ip_forward
```

② 定义规则使用的相关变量：

```
# 定义变量
IPT=/sbin/iptables
WEB_SERVER=210.10.18.89
FTP_SERVER=210.10.18.90
DNS_SERVER=210.10.18.91
SMTP_SERVER=210.10.18.92
PROTECT_DOMAIN="210.10.18.0/24"
```

③ 刷新基本的链规则，并禁止转发任意包：

```
$IPT_LIST -F
$IPT_LIST -P FORWARD DROP
```

④ 设置有关保护服务器的包过滤规则，并且注意由于服务器/客户机交互是双向的，所以不仅仅要设置数据包出去的规则，还要设置数据包返回的规则，如下所示：

```
#保护 WWW 服务：服务端口为 80，采用 TCP 或 UDP 协议
#规则为：eth1=>允许目的为内部网 WWW 服务器的包
$IPT_LIST -A FORWORD -p tcp udp -d $WEB_SERVER -dport www -i eth1 -j ACCEPT

#保护 FTP 服务：服务端口为：命令端口 21，数据端口 20。FTP 服务采用 TCP 协议。
# 规则为：eth1=>允许目的为内部网 FTP 服务器的包
$IPT_LIST -A FORWORD -p tcp -d $FTP_SERVER -dport ftp -i eth1 -j ACCEPT

#保护 DNS 服务：DNS 端口 53，采用 TCP 协议或者 UDP 协议
#规则为：eth1=>允许目的为内部网 DNS 服务器的查询请求
$IPT_LIST -A FORWORD -p tcp udp -d $ DNS_SERVER -dport dns -i eth1 -j ACCEPT

#保护 SMTP 服务：SMTP 端口 25，采用 TCP 协议
#规则为：eth1=>允许目的为内部网 SMTP 服务器的 SMTP 请求
$IPT_LIST -A FORWORD -p tcp -d $ SMTP_SERVER -dport smtp -i eth1 -j ACCEPT
```

（4）执行脚本，使配置规则立刻生效：

```
# /etc/rc.d/firewall-for-networkservice
```

到现在为止，有关使用防火墙来保障网络服务的配置就完成了。通过执行上面的脚本，我们建立了一个相对完整的防火墙。由于该防火墙只对外开放了有限的几个端口，因此能够有效地限制除这几个服务之外的流量进入，从而达到保证安全的目的。当然，这个例子还非常简单，用户还可以在实际中通过针对具体服务细化防火墙配置，比如限制外部网络并发连接 WWW 服务的 TCP 请求个数、限制外网访问 FTP 服务器的 IP 地址列表等来进行。这里只是给出一个简要的示例，以达到抛砖引玉的目的。

## 13.4 使用 Iptables 完成 NAT 功能

### 13.4.1 NAT 简介

在传统的标准 TCP/IP 通信过程中，所有的路由器仅仅是充当一个中间人的角色，也就是通常所说的存储转发，路由器并不会对转发的数据包进行修改，更为确切地说，除了将源 MAC 地址换成自己的 MAC 地址以外，路由器不会对转发的数据包做任何修改。NAT（Network Address

Translation）恰恰是出于某种特殊需要而对数据包的源 IP 地址、目的 IP 地址、源端口、目的端口进行改写的操作。

一般来说，以下几种情况下需要做 NAT。

- 为用户群提供 Internet 接入服务。为了方便管理，ISP 分配给用户的 IP 地址都是伪 IP，但是部分用户要求建立自己的 WWW 服务器对外发布信息，这时候我们就可以通过 NAT 来提供这种服务了。我们可以在防火墙的外部网卡上绑定多个合法 IP 地址，然后通过 NAT 技术使发给其中某一个 IP 地址的包转发至内部某一用户的 WWW 服务器上，然后再将该内部 WWW 服务器响应包伪装成该合法 IP 发出的包。
- 使用拨号上网的网吧。因为只有一个合法的 IP 地址，必须采用某种手段让其他机器也可以上网，通常是采用代理服务器的方式，但是代理服务器，尤其是应用层代理服务器，只能支持有限的协议，如果过了一段时间后又有新的服务出来，则只能等待代理服务器支持该新应用的升级版本。如果采用 NAT 来解决这个问题，因为是在应用层以下进行处理，NAT 不但可以获得很高的访问速度，而且可以无缝地支持任何新的服务或应用。
- 还有一个方面的应用就是重定向，也就是当接收到一个包后，不是转发这个包，而是将其重定向到系统上的某一个应用程序。最常见的应用就是和 squid 配合使用成为透明代理，在对 http 流量进行缓存的同时，可以提供对 Internet 的无缝访问。

### 13.4.2　NAT 的原理

Linux 将 NAT 分成了两种类型，即源 NAT（SNAT）和目的 NAT（DNAT），顾名思义，所谓 SNAT 就是改变转发数据包的源地址，所谓 DNAT 就是改变转发数据包的目的地址。前面提到过，Netfilter 是 Linux 核心中的一个通用架构，它提供了一系列的“表”（tables），每个表由若干“链”（chains）组成，而每条链中可以有一条或数条规则（rule）组成。并且系统默认的表是“filter”。但是在使用 NAT 的时候，我们所使用的表不再是“filter”，而是“nat”表，所以我们必须使用“-t nat”选项来显式地指明这一点。因为系统默认的表是“filter”，所以在使用 filter 功能时，我们没有必要显式地指明“-t filter”。同 filter 表一样，nat 表也有三条默认的“链”（chains），这三条链也是规则的容器，它们分别如下。

- PREROUTING：可以在这里定义进行目的 NAT 的规则，因为路由器进行路由时只检查数据包的目的 IP 地址，所以为了使数据包得以正确路由，我们必须在路由之前就进行目的 NAT。
- POSTROUTING：可以在这里定义进行源 NAT 的规则，系统在决定了数据包的路由以后再执行该链中的规则。
- OUTPUT：定义对本地产生的数据包的目的 NAT 规则。

如前所述，在使用 Iptables 的 NAT 功能时，我们必须在每一条规则中使用-t nat 显式地指明使用 nat 表。然后使用以下选项。

1）对规则的操作

- 加入（append）一个新规则到一个链（-A）的最后。
- 在链内某个位置插入（insert）一个新规则（-I），通常是插在最前面。
- 在链内某个位置替换（replace）一条规则（-R）。

- 在链内某个位置删除（delete）一条规则（-D）。
- 删除（delete）链内的第一条规则（-D）。

2）指定源地址和目的地址

通过--source/--src/-s 来指定源地址（这里的/表示或者的意思，下同），通过--destination/--dst/-s 来指定目的地址。可以使用以下四种方法来指定 IP 地址。

- 使用完整的域名，如 www.tsinghua.edu。
- 使用 IP 地址，如 172.168.92.10。
- 用 IP 地址/子网掩码指定一个网络地址，如 192.168.1.0/255.255.255.0。
- 用 IP 地址/子网掩码的位数指定一个网络地址，如 192.168.1.0/24，这里的 24 表明了子网掩码的有效位数，这是 Linux 环境中通常使用的表示方法。默认的子网掩码数是 32，也就是说，指定 172.168.92.10 等效于 172.168.92.10/32。

3）指定网络接口

可以使用--in-interface/-i 或--out-interface/-o 来指定网络接口。从 NAT 的原理可以看出，对于 PREROUTING 链，我们只能用-i 指定进来的网络接口；而对于 POSTROUTING 和 OUTPUT 我们只能用-o 指定出去的网络接口。

4）指定协议及端口

可以通过--protocol/-p 选项来指定协议，如果是 UDP 和 TCP 协议，还可使用--source-port/--sport 和 --destination-port/--dport 来指明端口。

### 13.4.3　NAT 的具体使用

#### 1．源地址 NAT

1）标准的 SNAT

SNAT 的目的是进行源地址转换，应用于 POSTROUTING 规则链。在路由决定之后应用 SNAT 与出站接口相关，而不是入站接口。语法如下：

```
 iptables -t nat -A POSTROUTING -o <outgoing interface> -j SNAT --to-source
<address>[-<address>][:port-port]
```

2）MASQUERADE 源 NAT

MASQUERADE 没有选项来指定在 NAT 设备上使用的特定源地址，使用的源地址就是出站接口的地址。

```
iptables -t nat -A POSTROUTING -o <outgoing interface> -j MASQUERADE [--to-ports
<port>[-port]]
```

举一个简单的例子：更改所有来自 192.168.1.0/24 的数据包的源 IP 地址为 1.2.3.4：

```
#iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o eth0 -j SNAT --to 1.2.3.4
```

这里需要注意的是，系统在路由及过滤等处理直到数据包要被送出时才进行 SNAT。

另外，有一种 SNAT 的特殊情况是 IP 欺骗，也就是所谓的 Masquerading，通常建议在使用拨号上网的时候使用，或者说在合法 IP 地址不固定的情况下使用。比如：

```
# iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

可以看出，这时候我们没有必要显式地指定源 IP 地址等信息。

### 2．目的地址 NAT

目的地址 NAT 有两种形式：DNAT 和 REDIRECT。REDIRECT 是目的地址转换的特殊形式，将数据包重定向到 NAT 设备的输入或回环接口。目的地址 NAT 应用于 nat 表的 PREROUTING 和 OUTPUT 规则链，在做出路由决定前对目的地址进行修改。在 PREROUTING 中，DNAT 和 REDIRECT 规则与用来接收通过本地路由转发或送到主机的入站接口的数据包的入站接口有关。在 OUTPUT 中，DNAT 和 REDIRECT 规则用来处理来自 NAT 主机本身生成的出站数据包。

1）标准目的地址 NAT（DNAT）

```
    iptables -t nat -A PREROUTING -i <incoming interface> -j DNAT --to-destination
<address>[-<address>][:port-port]
    iptables -t nat -A OUTPUT -o <outgoing interface> -j DNAT --to-destination
<address>[-<address>][:port-port]
```

目的地址用来替换数据包中的原始目的地址和多位本地服务器地址。

2）REDIRECT

```
    iptables -t nat -A PREROUTING -i <incoming interface> -j REDIRECT [--to-ports
<port>[-port]]
    iptables -t nat -A OUTPUT -o <outgoing interface> -j REDIRECT [--to-ports
<port>[-port]]
```

REDIRECT 重定向数据包到执行 REDIRECT 操作的那台主机。

举个简单的例子：更改所有来自 192.168.1.0/24 的数据包的目的 IP 地址为 1.2.3.4：

```
    #iptables -t nat -A PREROUTING -s 192.168.1.0/24 -i eth1 -j DNAT --to 1.2.3.4
```

这里需要注意的是，系统是先进行 DNAT，然后才进行路由及过滤等操作。有一种 DNAT 的特殊情况是重定向，也就是所谓的 Redirection，这时候就相当于将符合条件的数据包的目的 IP 地址改为数据包进入系统时的网络接口的 IP 地址。通常是在与 squid 配置形成透明代理时使用，假设 squid 的监听端口是 3128，我们可以通过以下语句将来自 192.168.1.0/24、目的端口为 80 的数据包重定向到 squid 监听端口：

```
    #iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.1.0/24 --dport 80 -j
REDIRECT --to-port 3128
```

### 3．一个具体的应用实例

为了更加系统和全面地介绍 NAT 的使用，下面举一个实际的例子来进行说明。假设有一家 ISP 提供园区 Internet 接入服务，为了方便管理，该 ISP 分配给园区用户的 IP 地址都是私网 IP，通过该私网 IP 用户无法向外发布信息。但是，部分用户要求建立自己的 WWW 服务器对外发布信息。我们可以在防火墙的外部网卡上绑定多个合法 IP 地址，然后通过 IP 映射使发给其中某一个 IP 地址的包转发至内部某一用户的 WWW 服务器上，然后再将该内部 WWW 服务器响应包伪装成该合法 IP 发出的包。

具体的 IP 分配如下。

（1）该 ISP 分配给 A 单位 WWW 服务器的 IP 如下。

私网 IP：172.168.92.100

公网 IP：210.95.33.100

（2）该 ISP 分配给 B 单位 WWW 服务器的 IP 如下。

私网 IP：172.168.92.200

公网 IP：210.95.33.200

（3）Linux 防火墙的 IP 地址如下。

内网接口 eth1：172.168.92.10

外网接口 eth0：210.95.33.1

然后，我们需要进行如下步骤的操作。

（1）将分配给 A、B 单位的真实 IP 绑定到防火墙的外网接口，以 root 权限执行以下命令：

```
#ifconfig eth0 add 210.95.33.100 netmask 255.255.255.0
#ifconfig eth0 add 210.95.33.200 netmask 255.255.255.0
```

（2）成功升级内核后安装 Iptables，然后执行以下脚本载入相关模块：

```
modprobe ip_tables
modprobe ip_nat_ftp
```

（3)对防火墙接收到的目的 IP 为 210.95.33.100 和 210.95.33.200 的所有数据包进行目的 NAT（DNAT）：

```
#iptables -A PREROUTING -i eth0 -d 210.95.33.100 -j DNAT --to 172.168.92.100
#iptables -A PREROUTING -i eth0 -d 210.95.33.200 -j DNAT --to 172.168.92.200
```

（4）对防火墙接收到的源 IP 地址为 172.168.92.100 和 172.168.92.200 的数据包进行源 NAT（SNAT）：

```
#iptables -A POSTROUTING -o eth0 -s 172.168.92.100 -j SNAT --to 210.95.33.100
#iptables -A POSTROUTING -o eth0 -s 172.168.92.200 -j SNAT --to 210.95.33.200
```

这样，所有目的 IP 为 210.95.33.100 和 210.95.33.200 的数据包都将分别被转发给 172.168.92.100 和 172.168.92.200；而所有来自 172.168.92.100 和 172.168.92.200 的数据包都将分别被伪装成 210.95.33.100 和 210.95.33.200，从而也就实现了 IP 映射。

## 13.5 防火墙与 DMZ 的配合使用

### 13.5.1 DMZ 原理

DMZ 是英文 demilitarized zone 的缩写，中文名称为“隔离区”，也称“非军事化区”。它是为了解决安装防火墙后外部网络不能访问内部网络服务器的问题，而设立的一个非安全系统与安全系统之间的缓冲区，这个缓冲区位于企业内部网络和外部网络之间的小网络区域内，在这个小网络区域内可以放置一些必须公开的服务器设施，如企业 Web 服务器、FTP 服务器和论坛等。另一方面，通过这样一个 DMZ 区域，更加有效地保护了内部网络，因为这种网络部署，比起一

般的防火墙方案，对攻击者来说又多了一道关卡，其网络结构如图 13-5 所示。网络设备开发商利用这一技术，开发出了相应的防火墙解决方案。DMZ 通常是一个过滤的子网，它在内部网络和外部网络之间构造了一个安全地带。

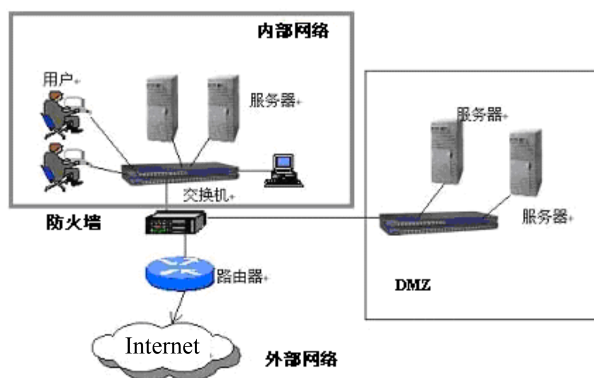



图 13-5　DMZ 示意图

DMZ 防火墙方案为要保护的内部网络增加了一道安全防线，通常认为是非常安全的。同时它提供了一个区域放置公共服务器，从而又能有效地避免一些互联应用需要公开，而与内部安全策略相矛盾的情况发生。在 DMZ 区域中通常包括堡垒主机、Modem 池，以及所有的公共服务器，但要注意的是电子商务服务器只能用作用户连接，真正的电子商务后台数据需要放在内部网络中。在这个防火墙方案中，包括两个防火墙，外部防火墙抵挡外部网络的攻击，并管理所有内部网络对 DMZ 的访问。内部防火墙管理 DMZ 对于内部网络的访问。内部防火墙是内部网络的第三道安全防线（前面有了外部防火墙和堡垒主机），当外部防火墙失效的时候，它还可以起到保护内部网络的功能。而局域网内部，对于 Internet 的访问由内部防火墙和位于 DMZ 的堡垒主机控制。在这样的结构里，一个黑客必须通过三个独立的区域（外部防火墙、内部防火墙和堡垒主机）才能够到达局域网，攻击难度大大增加相应内部网络的安全性也就大大加强，但投资成本也是最高的。

### 13.5.2　构建 DMZ

#### 1．构建原则

Linux 从 2.4 内核开始，正式使用 Iptables 来代替以前的 Ipfwadm 和 Ipchains，实现管理 Linux 的包过滤功能。Linux 的包过滤通过一个叫 Netfilter 的内核部件来实现。Netfilter 内建了三个表，其中默认表 filter 中又包括 3 个规则链，分别是负责外界流入网络接口的数据过滤的 INPUT 链、负责对网络接口输出的数据进行过滤的 OUTPUT 链，以及负责在网络接口之间转发数据过滤的 FORWARD 链。

要在 Linux 系统中构建一个带 DMZ 的防火墙，需要利用对这些链的设定完成。首先要对从连接外部网络的网卡（eth0）上流入的数据进行判断，这是在 INPUT 链上完成。如果数据的目标地址属于 DMZ 网段，就要将数据转发到连接 DMZ 网络的网卡（eth1）上；如果是内部网络的地址，就要将数据转发到连接内部网络的网卡（eth2）上。表 13-4 显示了各个网络之间的访问许可关系。

表 13-4　DMZ 和内外网的访问关系

| | 内　网 | 外　网 | DMZ |
|---|---|---|---|
| 内网 | / | 允许 | 允许 |
| 外网 | 禁止 | / | 允许 |
| DMZ | 禁止 | 禁止 | / |

根据表 13-4，可以明确以下六条访问控制策略。

- 内网可以访问外网。内网的用户显然需要自由地访问外网。在这一策略中，防火墙需要进行源地址转换。
- 内网可以访问 DMZ。此策略是为了方便内网用户使用和管理 DMZ 中的服务器。
- 外网不能访问内网：内网中存放的是公司内部数据，这些数据不允许外网的用户进行访问。
- 外网可以访问 DMZ。DMZ 中的服务器本身就是要给外界提供服务的，所以外网必须可以访问 DMZ。同时，外网访问 DMZ 需要由防火墙完成对外网地址到服务器实际地址的转换。
- DMZ 不能访问内网。很明显，如果违背此策略，则当入侵者攻陷 DMZ 时，就可以进一步进攻到内网的重要数据。
- DMZ 不能访问外网。此条策略在有的情况下可能会有例外，比如 DMZ 中放置邮件服务器时，就需要访问外网，否则将不能正常工作。

### 2. DMZ 的具体实现

根据以上访问控制策略可以设定 Linux 防火墙的过滤规则。下面将在一个虚构的网络环境中，探讨如何根据以上六条访问控制策略建立相应的防火墙过滤规则。这里的讨论和具体应用会有所区别，不过这种讨论将有助于实际应用。用户在实际应用时可根据具体情况进行设置。该虚拟环境的网络拓扑如图 13-6 所示。

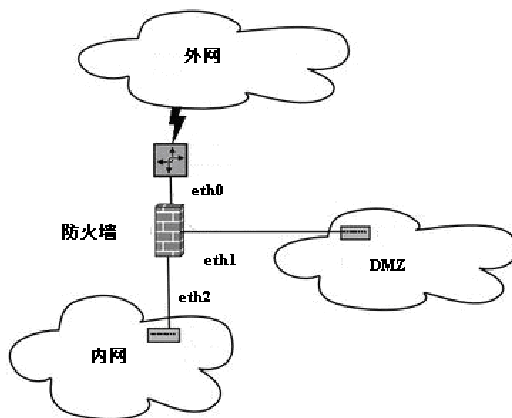



图 13-6　DMZ 网络拓扑图

如图 13-6 所示，路由器连接 Internet 和防火墙。作为防火墙的 Linux 服务器使用三块网卡：网卡 eth0 与路由器相连，网卡 eth1 与 DMZ 区的 Hub 相连，网卡 eth2 与内网 Hub 相连。作为一个抽象的例子，我们用“内网地址”来代表“192.168.1.0/24”之类的具体数值。同理还有“外网地址”和“DMZ 地址”。

对于防火墙，原则之一就是默认禁止所有数据通信，然后再打开必要的通信。所以在防火墙脚本的最初，需要清空系统原有的规则，然后将 INPUT、OUTPUT、FORWARD 的默认规则设置为丢弃所有数据包。

1）防火墙基本设置

对应的防火墙脚本片段如下：

```
# Flush out the tables and delete alluser-definedchains
/sbin/iptables -F
/sbin/iptables -X
/sbin/iptables -t nat -F
/sbin/iptables -t nat -X
```

```
# Drop every packet
/sbin/iptables -P INPUT DROP
/sbin/iptables -P OUTPUT DROP
/sbin/iptables -P FORWARD DROP
```

2）六种策略的具体实现

（1）内网可以访问外网。

对应的防火墙脚本片段如下：

```
/sbin/iptables -t nat -A POSTROUTING -s [内网地址] -d [外网地址] -oeth0-j SNAT
--to [NAT 的真实 IP]
```

当数据从连接外网的eth0流出时，要将来自内网的数据包的源地址改成Internet上的真实IP，这样才能和外网的主机进行通信。“NAT 的真实 IP”表示分配给 NAT 用户的真实 IP，有几个就写几个，以空格分开，但至少要写一个。

（2）内网可以访问 DMZ。

对应的防火墙脚本片段如下：

```
/sbin/iptables -A FORWARD -s [内网地址] -d [DMZ 地址] -i eth2-jACCEPT
```

以上命令允许所有来自内网、目的地为 DMZ 的数据包通过。

（3）外网不能访问内网。

对应的防火墙脚本片段如下：

```
/sbin/iptables -t nat -A PREROUTING -s [外网地址] -d [内网地址] -i eth0-jDROP
```

以上命令将来自外网、去往内网的数据包全部丢弃。

（4）外网可以访问 DMZ。

为了保护 DMZ 中的服务器，外网对 DMZ 的访问也要加以限制。通常的思路是，只允许外网访问 DMZ 中服务器所提供的特定服务，比如 HTTP。

对应的防火墙脚本片段如下：

```
/sbin/iptables -t nat -A PREROUTING -p tcp --dport 80-d[分配给 HTTP 服务器的
Internet 上的真实 IP] -s [外网地址] -i eth0 -j DNAT--to[HTTP 服务器的实际 IP]
/sbin/iptables -A FORWARD -p tcp -s [外网地址] -d [HTTP 服务器的实际 IP]-ieth0
--dport 80 -j ACCEPT
/sbin/iptables -A FORWARD -p tcp -d [外网地址] -s [HTTP 服务器的实际 IP]-ieth1
--sport 80 ! --syn -j ACCEPT
/sbin/iptables -t nat -A PREROUTING -s [外网地址] -d [DMZ 地址] -ieth0-j DROP
```

该防火墙脚本片段将开放 HTTP 服务，使得只有访问 DMZ 中 HTTP 服务的数据包才能通过防火墙。

（5）DMZ 不能访问内网。

对应的防火墙脚本片段如下：

```
/sbin/iptables -A FORWARD -s [DMZ 地址] -d [内网地址] -i eth1 -jDROP
```

以上命令将丢弃所有从 DMZ 到内网的数据包。

（6）DMZ 不能访问外网。

对应的防火墙脚本片段如下：

```
/sbin/iptables -t nat -A POSTROUTING -p tcp --dport 25 -d[外网地址]-s [邮件服务器的 IP] -o eth0 -j SNAT --to[分配给 SMTP 服务器的 Internet 上的真实 IP]
/sbin/iptables -A FORWARD -p tcp -s [邮件服务器的 IP] -d [外网地址] -ieth1--dport 25 -j ACCEPT
/sbin/iptables -A FORWARD -p tcp -d [邮件服务器的 IP] -s [外网地址]-ieth0--sport 25 ! --syn -j ACCEPT
```

以上命令先允许 DMZ 中邮件服务器连接外网的 SMTP 服务端口（25），然后禁止其他从 DMZ 发往外网的数据包。

针对以上基本策略列举了实现它们的基本规则。在实际应用中，需要根据具体情况进行设置。只要设置得当，Linux 也能成为很好的防火墙。需要补充的是，无论何种防火墙都只能提供有限的保护。设置好防火墙不等于网络就是安全的，关键在于综合运用各种安全手段。

## 13.6 防火墙的实际安全部署建议

防火墙在实际的部署应用过程当中，经常部署在网关的位置，也就是在网内和网外的一个“中间分隔点”上，而就是在这样一个部署环境中，也还存在着多种方式，且存在着许多“陷阱”，下面进行详细分析。

### 13.6.1 方案一：错误的防火墙部署方式

传统的防火墙部署方式可能所有人都认为非常简单，将防火墙部署于外部网络和内部网络之间。这个思路如果在内部网络中存在共享资源（比如说 FTP 服务器和 Web 服务器）的话，那么这将是一个非常危险的部署方式，如图 13-7 所示。理由其实非常简单，一旦这些共享服务器被黑客攻击和安装木马渗透病毒的话，那么内部网络的客户端及其资源将没有任何安全可言。因为在这种情况下，木马和病毒已经在内网中存在，而客户端和共享资源服务器在同一个网段，这无异于内网的安全隐患，防火墙对此无能为力，从而也失去了部署的意义。

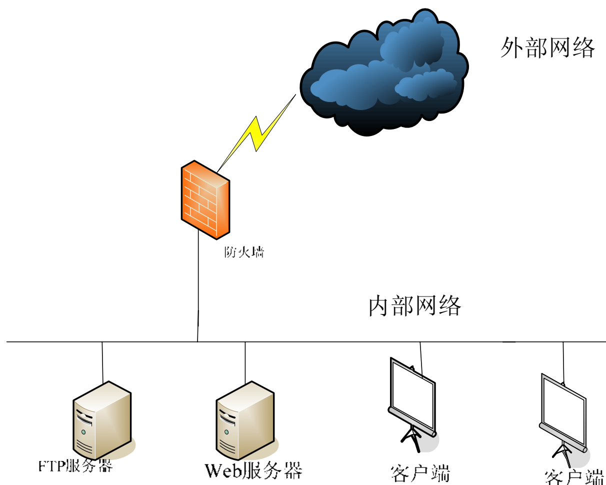



图 13-7 错误的防火墙部署方式

### 13.6.2　方案二：使用 DMZ

目前一个比较流行和正确的做法就是采用 13.5 节介绍的采用 DMZ 防火墙部署方式，如图 13-8 所示。也就是在防火墙上多加一块网卡，把提供对外服务的服务器和内网的客户端严格地隔离开来，这样，即使有安全风险和漏洞在 DMZ 中出现，由此对内部网络造成的危害也可以得到很好的控制，从而避免了方案一的缺点。

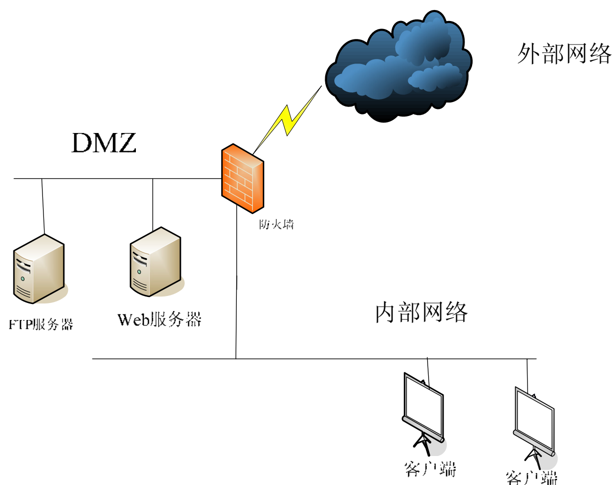



图 13-8　使用 DMZ 的防火墙部署方式

### 13.6.3　方案三：使用 DMZ+二路防火墙

为了加强方案二中防火墙的安全强度，目前有些企业将图 13-8 所示的架构优化成图 13-9 所示的架构，也就是使用 DMZ+二路防火墙。另外，在此结构中选用防火墙，应尽量采用两家不同公司的产品，这样才有利于发挥这种架构的优势。

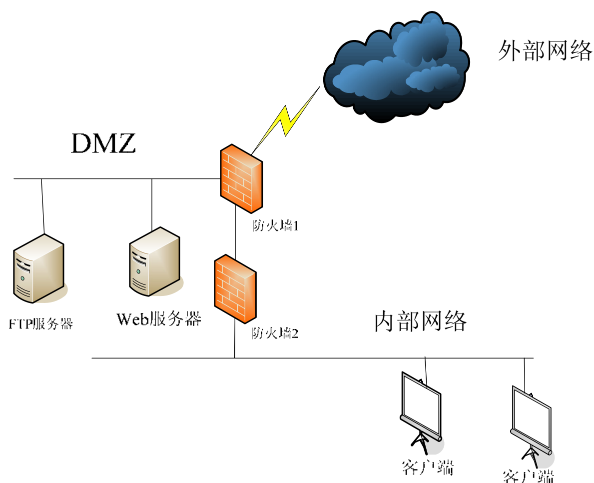



图 13-9　使用 DMZ+二路防火墙的部署方式

### 13.6.4　方案四：通透式防火墙

在前面的几种方案中，防火墙本身就是一个路由器，在使用的过程中用户必须慎重地考虑到路由的问题。如果网络环境非常复杂或者是需要进行调整，则相应的路由需要进行变更，维护和操作起来有一定的难度和工作量。

如图 13-10 所示的通透式防火墙则可以比较好地解决上述问题，该类防火墙是一个桥接设备，并且在桥接设备上赋予了过滤的能力。由于桥接设备工作在 OSI 模型的第二层（也就是数据

链路层），所以不会有任何路由的问题。并且，防火墙本身也不需要指定 IP 地址，因此，这种防火墙的部署能力和隐密能力都相当强，从而可以很好地应对黑客对防火墙自身的攻击，因为黑客很难获得可以访问的 IP 地址。

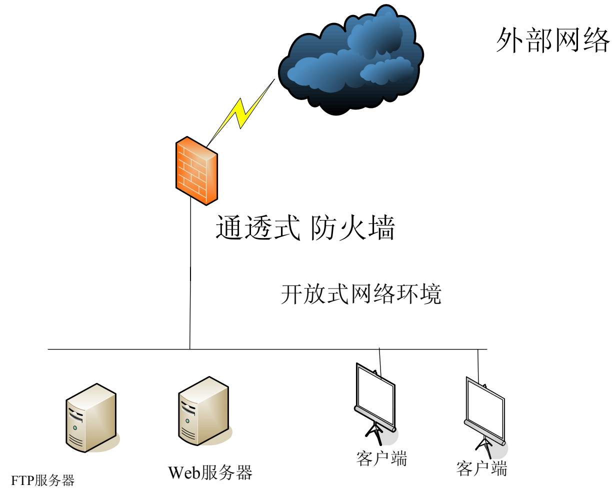



图 13-10　通透式防火墙部署方式

# 第 14 章

# 铜墙铁壁：企业立体式入侵检测及防御

## 本章导读

在实际的入侵检测及防御体系的构建中，有的以网络为主，进行网络威胁的发现和封堵；有的以主机防御为主，主要保证主机不遭受入侵。如果仅针对其中一方面进行构建的话，则会存在偏差，对于企业500强企业来说，建议综合多方面的信息，进行纵深的综合性防御，这样才能起到很好的效果。

在开源系统中，例如Linux操作系统中，从应用到内核层面上提供了三种入侵检测系统来对网络和主机进行防御，它们分别是网络入侵检测系统Snort、主机入侵检测系统 LIDS 以及分布式入侵检测系统SnortCenter。其中：

- Snort专注于在网络层面进行入侵检测。
- LIDS 则侧重于在主机层面进行入侵检测和防御。
- SnortCenter 则是为了在分布式环境中提升入侵检测的实时性和准确性的一种分布式检测机制。

本章将依据这三种入侵系统，就如何配置、部署和使用对它们进行详细介绍。

## 14.1 入侵检测技术简介

本节内容参考 2.4.1 节。

## 14.2 网络入侵检测及防御：Snort

### 14.2.1 安装 Snort

说起 Snort，很多熟悉它的用户首先会想到那个可爱的喷气猪造型。没错，那就是 Snort 的吉祥物，如图 14-1 所示。一般市面上的介绍都把 Snort 说得好像是纯个人开发的开源玩具一样，其实它很实用，而且有一家名叫 Sourefire 的上市公司在负责研发和升级工作。这个工具本身是免费的，可以在任何操作系统上运行，包括 Windows 和 Linux，但是个人认为它的配置过程还是比较复杂的。

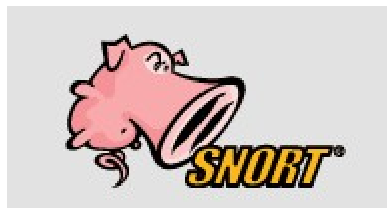



图 14-1 Snort 吉祥物

Snort 有三种工作模式：嗅探器、数据包记录器和网络入侵检测系统。嗅探器模式仅仅是从网络上读取数据包并作为连续不断的流显示在终端上；数据包记录器模式把数据包记录到硬盘上；网络入侵检测模式是最复杂的，我们可以让 Snort 分析网络数据流以匹配用户定义的一些规则，并根据检测结果采取一定的动作。

虽然这几年 Snort 一直在推陈出新，但是安装 Snort 总体来说还是具有一定技术含量的。安装命令如下，首先需要确定机器上已经安装了 zlib、libpcap、MySQL、Apache、PHP 和其他多种底层开发工具和开发库：

```
//安装 Snort RPM 包
[root@localhost ~]# rpm -ivh snort-2.8.6-1.RH5.i386.rpm
Preparing...                ########################################### [100%]
   1:snort                  ########################################### [100%]

//安装 MySQL 数据库
[root@localhost ~]# rpm -ivh snort-mysql-2.8.6-1.RH5.i386.rpm
Preparing...                ########################################### [100%]
   1:snort-mysql            ########################################### [100%]
```

### 14.2.2 配置 Snort

配置 Snort 的步骤比较多，主要有以下几个步骤。

（1）修改 Snort 的配置文件 snort.conf（见图 14-2）：

```
[root@localhost ~]# vim /etc/snort/snort.conf
```

```
#--------------------------------------------------
#   VRT Rule Packages Snort.conf
#
#   For more information visit us at:
#     http://www.snort.org                   Snort Website
#     http://vrt-sourcefire.blogspot.com/    Sourcefire VRT Blog
#
#     Mailing list Contact:      snort-sigs@lists.sourceforge.net
#     False Positive reports:    fp@sourcefire.com
#     Snort bugs:                bugs@snort.org
#
#     Compatible with Snort Versions:
#     VERSIONS : 2.9.0.3
#
#     Snort build options:
#     OPTIONS : --enable-ipv6 --enable-gre --enable-mpls --enable-targetbased --enable-decoder-preprocessor-rules --enable-ppm --enable-perfprofiling --enable-zlib --enable-active-response --enable-no
malizer --enable-reload --enable-react --enable-flexresp3
#--------------------------------------------------

###################################################
# This file contains a sample snort configuration.
# You should take the following steps to create your own custom configuration:
#
#  1) Set the network variables.
#  2) Configure the decoder
#  3) Configure the base detection engine
#  4) Configure dynamic loaded libraries
#  5) Configure preprocessors
#  6) Configure output plugins
#  7) Customize your rule set
#  8) Customize preprocessor and decoder rule set
#  9) Customize shared object rule set
###################################################

###################################################
# Step #1: Set the network variables.  For more information, see README.variables
###################################################

# Setup the network addresses you are protecting
ipvar HOME_NET any
```

图 14-2　修改配置文件

（2）配置规则：在配置文件中有两类规则：alert 和 log，我们在下面加一条语句即可。

```
# output database: alert, <db_type>, user=<username> password=<password> test dbname=<name> host=<hostname>

# output database: log, <db_type>, user=<username> password=<password> test dbname=<name> host=<hostname>
```

比如我们加这么一条语句：

```
output database: log, mysql, user=root password=xxxxxx dbname=snort host=localhost
```

（3）使规则生效，把下面语句前面的“#”去掉即可：

```
# include $RULE_PATH/web-attacks.rules
# include $RULE_PATH/backdoor.rules
# include $RULE_PATH/shellcode.rules
# include $RULE_PATH/policy.rules
# include $RULE_PATH/porn.rules
# include $RULE_PATH/info.rules
# include $RULE_PATH/icmp-info.rules
 #include $RULE_PATH/virus.rules
# include $RULE_PATH/chat.rules
 #include $RULE_PATH/multimedia.rules
# include $RULE_PATH/p2p.rules
```

（4）创建 Snort 的数据库：

```
[root@localhost ~]# mysql -u root -p
Enter password:
```

```
## 输入密码，登录 MySQL，创建 Snort 的数据库

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3 to server version: 5.0.22

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database snort;
Query OK, 1 row affected (0.00 sec)

mysql> use snort;
Database changed

mysql> grant create, insert, select ,delete,update on snort.* to snort;
Query OK, 0 rows affected (0.01 sec)

mysql> grant create,insert,select,delete,update on snort.* to snort@localhost;
Query OK, 0 rows affected (0.00 sec)

mysql> set password for 'snort'@'localhost' = password('123456');
Query OK, 0 rows affected (0.00 sec)

mysql> set password for 'snort'@'%' = password('123456');
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

具体步骤如图 14-3 所示。

图 14-3　创建 Snort 的数据库

```
mysql> use snort;
Database changed
mysql> source /usr/share/snort-2.8.6/schemas/create_mysql
Query OK, 0 rows affected (0.00 sec)
Query OK, 1 row affected (0.04 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.01 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.01 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.01 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.01 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 1 row affected (0.01 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 0 rows affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
Query OK, 1 row affected (0.00 sec)
mysql>
## 这样 Snort 数据库里面就已经导入了数据表了
mysql> quit
Bye
## 到这里，Snort 数据库就建立好了
```

（5）创建 Snort 访问的用户：

```
[root@localhost schemas]# htpasswd -c /etc/httpd/conf/htpasswd apacheman
New password:
Re-type new password:
Adding password for user apacheman
```

```
//配置 httpd，并开启该服务
 [root@localhost schemas]# vim /etc/httpd/conf/httpd.conf
[root@localhost schemas]# chown apache.apache /etc/httpd/conf/htpasswd
[root@localhost schemas]# etc/init.d/httpd restart

//拷贝 Snort 的规则库
tar zxvf snortrules-snapshot-2.8.tar.gz

cp rules/* /etc/snort/rules
把规则库都复制到 Snort 的 rules 文件夹下。

//显示 Snort 规则库
[root@localhost base]# cd /var/www/html
[root@localhost base]# tar zxvf adodb465.tgz

[root@localhost html]# mv jpgraph-2.1.1 jpgraph
[root@localhost html]# mv base-1.2.6 base
[root@localhost html]# cd base
[root@localhost base]# ls
admin                  base_graph_form.php   base_qry_sqlcalls.php  contrib
base_ag_common.php     base_graph_main.php   base_stat_alerts.php   docs
base_ag_main.php       base_hdr1.php         base_stat_class.php    help
base_common.php        base_hdr2.php         base_stat_common.php   images
base_conf.php.dist     base_main.php         base_stat_ipaddr.php   includes
base_db_common.php     base_maintenance.php  base_stat_iplink.php   index.php
base_db_setup.php      base_payload.php      base_stat_ports.php    languages
base_denied.php        base_qry_alert.php    base_stat_sensor.php   scripts
base_footer.php        base_qry_common.php   base_stat_time.php     setup
base_graph_common.php   base_qry_form.php    base_stat_uaddr.php    sql
base_graph_display.php  base_qry_main.php    base_user.php          styles

 [root@localhost base]#  cp base_conf.php.dist base_conf.php

vim base_conf.php
```

```
$DBlib_path = "/var/www/html/adodb";
$alert_dbname   = 'snort';
$alert_host     = 'localhost';
$alert_port     = '';
$alert_user     = 'snort';
$alert_password = '123456';

/* Archive DB connection parameters */
$archive_exists   = 0; # Set this to 1 if you have an archive DB
$archive_dbname   = 'snort';
$archive_host     = 'localhost';
$archive_port     = '';
$archive_user     = 'snort';
$archive_password = '123456';

$ChartLib_path = "/var/www/html/jpgraph/src";
```

配置的部分结果如图 14-4 所示。

（6）启动 MySQL 数据库：

```
[root@localhost base]# service mysqld start
 [root@localhost base]# service httpd start
```

如图 14-5 和图 14-6 所示。

```
$DBlib_path = '/var/www/html/adodb';

/* The type of underlying alert database
 *
 *  MySQL         : 'mysql'
 *  PostgresSQL   : 'postgres'
 *  MS SQL Server : 'mssql'
 *  Oracle        : 'oci8'
 */
$DBtype = 'mysql';

/* Alert DB connection parameters
 *   - $alert_dbname   : MySQL database name
 *   - $alert_host     : host on which the DB
 *   - $alert_port     : port on which to acc
 *   - $alert_user     : login to the databas
 *   - $alert_password : password of the DB u
 *
 *  This information can be gleaned from the
 *  output plugin configuration.
 */
$alert_dbname   = 'snort';
$alert_host     = 'localhost';
$alert_port     = '';
$alert_user     = 'snort';
$alert_password = '123456';
-- 插入 --
```

图 14-4　配置的部分结果

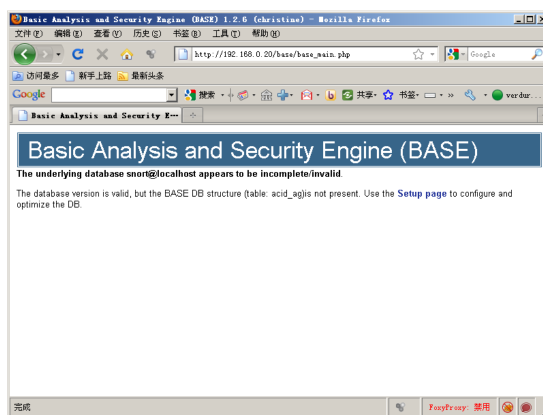



图 14-5　结果

进入 Setup page 链接。

单击 Create BASE AG 按钮，如图 14-7 所示。

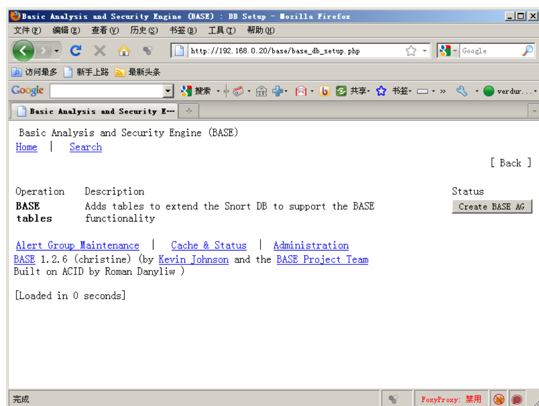


图 14-6　Setup page 链接

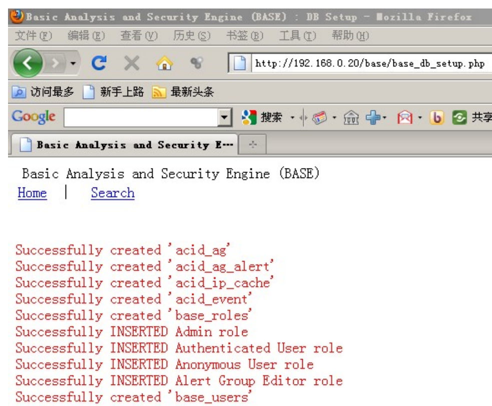


图 14-7　弹出页面

生成角色和用户等信息，如图 14-8 所示。

这个时候 Snort 还没开始工作，但是已经可以访问了。

如图 14-9 所示，用户名输入 apacheman，密码输入 xxxxxx。

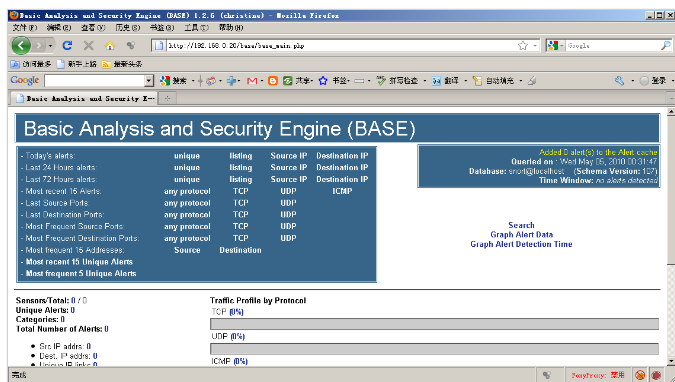


图 14-8　生成角色和用户信息



图 14-9　输入用户和密码

遇到图 14-10 所示的这种出错情况的话，很有可能是 acid_conf.php 文件没有修改。我们把文件中的“$alert_dbname= "snort_log"”；改成“$alert_dbname= "snort"; ”。

如图 14-11 所示，现在可以访问了。执行：/usr/sbin/snort -i eth0 -c /etc/snort/snort.conf –D。

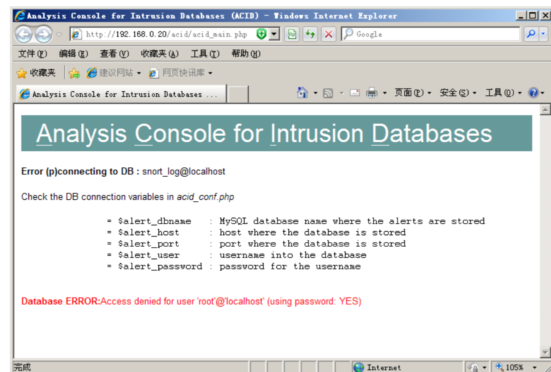


图 14-10　出错情况

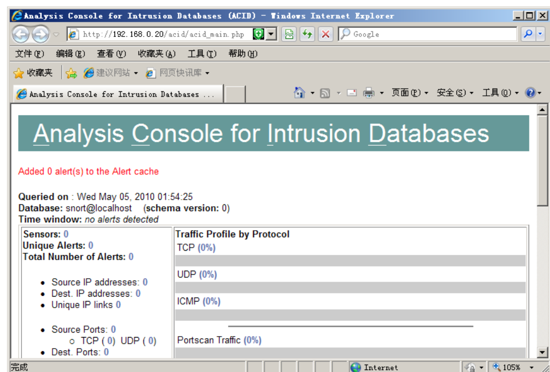


图 14-11　访问正常

看看效果，如图 14-12 所示。

现在 Snort 开始工作了。我们来看看曲线图的功能是否可以运行起来。

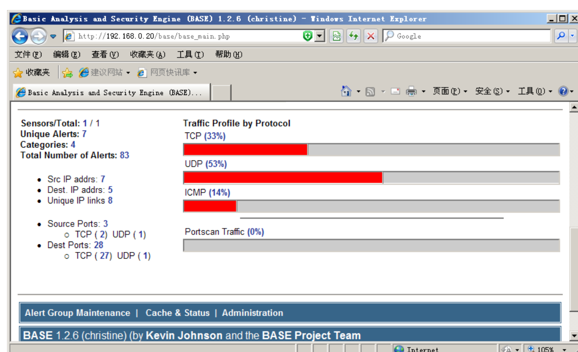


图 14-12　Snort 正常工作

如果出现图 14-13 这个错误。只要使用如下命令：

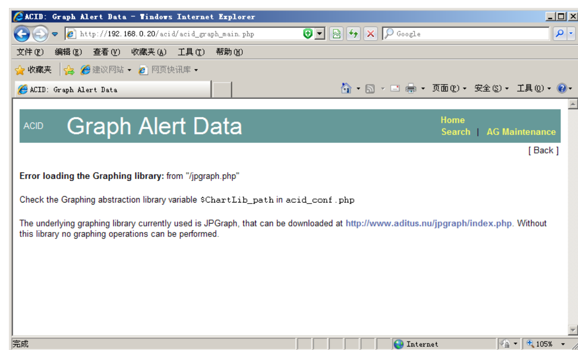


图 14-13　出错页面

```
[root@localhost acid]# vi acid_conf.php
```

修改成下面路径即可：

```
$ChartLib_path = "/var/www/html/jpgraph/src";
```

从图 14-14 可以看到，现在这个功能已经实现了。

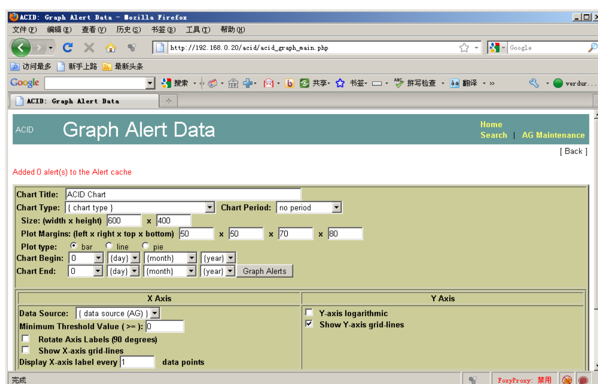


图 14-14　实现曲线图功能

## 14.3 编写 Snort 规则

Snort 规则一般是一行一条，规则头包含规则的动作、协议、源和目标 IP 地址、网络掩码，

以及目标端口信息；规则选项部分包含报警消息内容和要检查的包的具体部分。

### 14.3.1 规则动作

对于匹配特定规则的数据包，Snort 有以下 5 五种处理动作。

- alert：使用选择的报警方法生成一个警报，然后记录（log）这个包。
- log：记录这个包。
- pass：丢弃（忽略）这个包。
- activate：报警并且激活另一条 dynamic 规则。
- dynamic：保持空闲直到被一条 activate 规则激活，被激活后就作为一条 log 规则执行。

除了上述五个动作外，用户还可以定义自己的规则类型并且附加一条或者更多的输出模块给它，然后就可以使用这些规则类型作为 Snort 规则的一个动作。

举例如下：

```
    log tcp 192.168.0.0/24 any -> 192.168.1.0/24 22 (content:"|00 01 00 d2|"; msg:
" military incursion";)
```

上面规则的意思是：192.168.0.0 这个网段到 192.168.1.0 网段任何一个 22 端口的 TCP 流量，只要里面有符合的十六进制内容，就记录 military incursion。

### 14.3.2 协议

每条规则的第二项就是协议项。当前，Snort 能够分析的协议是：IP、TCP、UDP 和 ICMP。将来可能会提供对 ARP、ICRP、GRE、OSPF、RIP、IPX 等协议的支持。

### 14.3.3 IP 地址

规则头下面的部分就是 IP 地址和端口信息。关键词 any 可以用来定义任意的 IP 地址。Snort 不支持对主机名的解析，所以地址只能使用数字/CIDR 的形式。/24 表示一个 C 类网络；/16 表示一个 B 类网络；而/32 表示一台特定的主机地址。例如：

```
    192.168.1.0/24 表示从 192.168.1.1 到 192.168.1.255 的地址。
```

在规则中，可以使用否定操作符（negation operator）对 IP 地址进行操作。它告诉 Snort 除了列出的 IP 地址外，匹配所有的 IP 地址。否定操作符使用“!”表示。

```
    //下面这条规则中的 IP 地址表示：所有 IP 源地址不是内部网络的地址，而目的地址是内部网络地址。
    alert tcp !192.168.1.0/24 any -> 192.168.1.0/24 111（content:"|00 01 86 a5|";
msg:"external mountd access"）（使用 IP 地址否定操作符的规则）
```

也可以定义一个 IP 地址列表（IP list）。IP 地址列表的格式如下：

```
    [IP 地址 1/CIDR,IP 地址/CIDR,....]
```

需要注意的是：每个 IP 地址之间不能有空格。例如：

```
    alert tcp ![192.168.1.0/24,10.1.1.1.0/24] any ->[192.168.1.0/24,10.1.1.0/24]
111 (content:"|00 01 86 a5|"; msg:"external mountd access" )
```

### 14.3.4　端口号

在规则中，可以有几种方式来指定端口号，包括 any、静态端口号（static port）定义、端口范围，以及使用非操作定义。any 表示任意合法的端口号；静态端口号表示单个的端口号，例如：111（portmapper）、23（telnet）、80（http）等；使用范围操作符可以指定端口号范围，有几种方式来使用范围操作符“:”达到不同的目的，例如：

```
//记录来自任何端口，其目的端口号在 1～1024 之间的 UDP 数据包
log udp any any -> 192.168.1.0/24 1:1024
//记录来自任何端口，其目的端口号小于或者等于 6000 的 TCP 数据包
log tcp any any -> 192.168.1.0/24 :600
//记录源端口号小于等于 1024，目的端口号大于等于 500 的 TCP 数据包
log tcp any :1024 -> 192.168.1.0/24 500
```

还可以通过使用逻辑非操作符“!”对端口进行非逻辑操作（port negation）。逻辑非操作符可以用于其他的规则类型（除了 any 类型，道理很简单）。例如，如果要日志除了 X-Window 系统端口之外的所有端口，可以使用下面的规则：

```
log tcp any any -> 192.168.1.0/24 !6000:60 10 （对端口进行逻辑非操作）
```

### 14.3.5　方向操作符（direction operator）

方向操作符“->”表示数据包的流向。它左边是数据包的源地址和源端口，右边是目的地址和端口。此外，还有一个双向操作符“<>”，它使 Snort 对这条规则中，两个 IP 地址/端口之间双向的数据传输进行记录分析。

```
//下面的规则表示对一个 telnet 对话的双向数据传输进行记录：
log !192.168.1.0/24 any <> 192.168.1.0/24 23 （使用双向操作符的 Snort 规则）
```

### 14.3.6　activate/dynamic 规则

activate/dynamic 规则对扩展了 Snort 功能。使用 activate/dynamic 规则对，能够使用一条规则激活另一条规则。当一条特定的规则启动，如果想要 Snort 接着对符合条件的数据包进行记录时，使用 activate/dynamic 规则对非常方便。除了一个必选的选项 activates 外，激活规则（activate rule）非常类似于报警规则（alert rule）。动态规则（dynamic rule）和日志规则（log rule）也很相似，不过它需要一个选项：activated_by。动态规则还需要另一个选项：count。当一个激活规则启动，它就打开由 activate/activated_by 选项之后的数字指示的动态规则，记录 count 个数据包。下面是一条 activate/dynamic 规则对的规则：

```
activate tcp !$HOME_NET any -> $HOME_NET 143 （flagsA; content:"|E8C0FFFFFF
|◘in|; activates:1; <msg:"IMAP buffer overflow!" ）
（activate/dynamic 规则对 ）
```

上述规则使 Snort 在检测到 IMAP 缓冲区溢出时发出报警，并且记录后续的 50 个从 $HOME_NET 之外，发往$HOME_NET 的 143 号端口的数据包。如果缓冲区溢出成功，那么接下来 50 个发送到该网络同一个服务端口（这个例子中是 143 号端口）的数据包中，会有很重要

的数据，这些数据对以后的分析很有用处。

在 Snort 中有 23 个规则选项关键词，随着 Snort 不断地加入对更多协议的支持以及功能的扩展，会有更多的功能选项加入其中。这些功能选项可以以任意方式进行组合，对数据包进行分类和检测。现在，Snort 支持的选项包括：msg、logto、ttl、tos、id、ipoption、fragbits、dsize、flags、seq、ack、itype、icode、icmp_id、content、content-list、offset、depth、nocase、session、rpc、resp、react。每条规则中，各规则选项之间是逻辑与的关系。只有规则中的所有测试选项（例如：ttl、tos、id、ipoption 等）都为真，Snort 才会采取规则动作。

## 14.3.7 Snort 规则简单应用举例

### 1．拒绝服务攻击的规则

拒绝服务攻击的规则如下：

```
    alert udp any 19 <> any 7 (msg:"DOS UDP echo+chargen bomb"; flow:to_server;
metadata:policy security-ips drop; reference:cve,1999-0103; reference:cve,1999-0
    635; classtype:attempted-dos; sid:271; rev:8;)
```

### 2．SSH 攻击报警的规则

SSH 攻击报警的规则（经常被人用扫描器扫的人应该有体会）如下：

```
    alert tcp $HOME_NET 22 -> $EXTERNAL_NET any (msg:"ATTACK-RESPONSES successful
gobbles ssh exploit uname"; flow:from_server,established; content:"uname"; metad
    ata:policy  balanced-ips  drop,  policy  connectivity-ips  drop,  policy
security-ips drop; reference:bugtraq,5093; reference:cve,2002-0390; reference:
cve,2002-0639
    ; reference:nessus,11031; classtype:misc-attack; sid:1811; rev:12;)
```

### 3．对 ICMP 包回应的规则

对 ICMP 包回应的规则如下：

```
    alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP Echo Reply"; icode:0;
itype:0; classtype:misc-activity; sid:408; rev:5;)
```

图 14-15 所示是 ICMP 包回应截获数据的 Web 效果图。

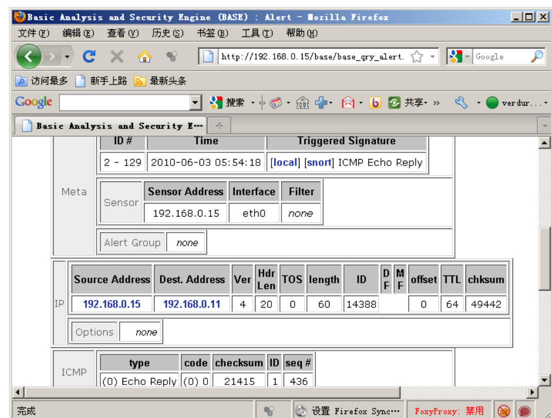



图 14-15 ICMP 包回应截获数据的 Web 效果图

#### 4．对 DNS 包回应的规则

对 DNS 包回应的规则如下：

```
    alert udp $EXTERNAL_NET 53 -> $HOME_NET any (msg:"DNS SPOOF query response with
TTL of 1 min. and no authority"; flow:to_client; content:"|81 80 00 01 00 01 00
00 00 00|"; content:"|C0 0C 00 01 00 01 00 00 00|<|00 04|"; metadata:policy
security-ips drop, service dns; classtype:bad-unknown; sid:254; rev:7;)
```

图 14-16 是 DNS 包回应截获数据的 Web 效果图。

上述规则都添加在/etc/snort/rules 文件里。给 rules 文件添加规则的效果图如图 14-17 所示。

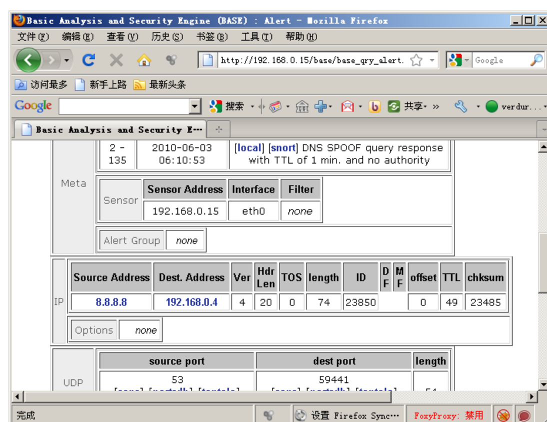



图 14-16　DNS 包回应截获数据的 Web 效果图

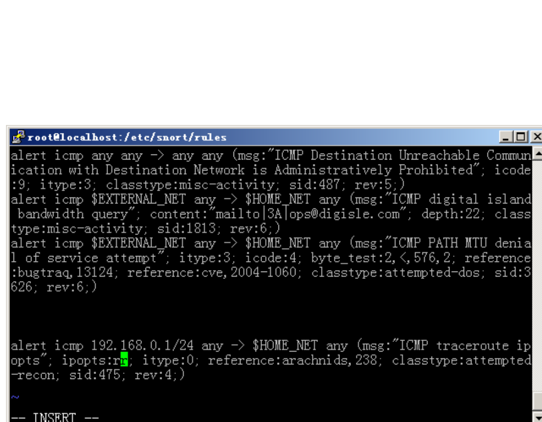



图 14-17　给 rules 文件添加规则的效果图

### 14.3.8　Snort 规则高级应用举例

上面介绍了如何编写 Snort 规则，下面再给出几个实际应用中的例子（几个对当前主要的攻击手段的检测的例子）来说明如何灵活、高效地应用 Snort 规则。

#### 1．检测尼姆达病毒

尼姆达是一个蠕虫病毒，在 2001 年的下半年曾经在互联网上肆虐。尼姆达的检测规则主要是在扫描的时候、运行的时候和传播的时候。由于该病毒的机理比较复杂，具体的分析过程不在这里详细阐述，可以给出以下的检测规则：

```
    preprocessor http_decode: 80 443 8080
    preprocessor minfrag: 128
    preprocessor portscan: 12.23.34.45/32 3 5 /var/log/snort_portscan.log
    preprocessor portscan-ignorehosts:
    alert tcp any any -> any 80 (msg:"W32/Nimda@mm WORM SCAN!"; flags:PA;
content:"/root.exe?/c+dir"; nocase;)
    alert tcp any any -> any 80 (msg:"W32/Nimda@mm WORM SCAN!"; flags:PA;
content:"/system32/cmd.exe?/c+dir"; nocase;)
    alert tcp any any -> any 80 (msg:"W32/Nimda@mm WORM TRANSFER!!"; flags:PA;
```

```
content:"/root.exe?/c+tftp%20-i"; nocase;)
    alert tcp any any -> any 80 (msg:"W32/Nimda@mm WORM TRANSFER!!"; flags:PA;
content:"/system32/cmd.exe?/c+tftp%20-i"; nocase;)
    alert tcp any any -> any 80 (msg:"W32/Nimda@mm WORM RUN!!!"; flags:PA;
content:"/scripts/Admin.dll"; nocase;)
    alert tcp any any -> any 80 (msg:"W32/Nimda@mm WORM RUN!!!"; flags:PA;
content:"/MSADC/Admin.dll"; nocase;)
    alert tcp any any -> any 80 (msg:"W32/Nimda@mm WORM RUN!!!"; flags:PA;
content:"/winnt/system32/Admin.dll"; nocase;)
    alert udp any 69 -> any any (msg:"W32/Nimda@mm WORM TRANSFER!!"; flags:PA;
content:"|15 90 AC 17 36 F7 D8 1B C0 5E 40 5B 5F C9 C2 04 00 55 8B EC 81 EC B0 00|";)
    alert tcp any 80 -> any any (msg:"W32/Nimda@mm WORM IN WEB SERVER!!"; flags:PA;
content:">window.open("readme.eml"";)
    alert tcp any any -> any 25 (msg:"W32/Nimda@mm WORM MAILSEND!!"; flags:PA;
content:"UgEAAI1F6Ild6FCNRfxQU2g/AA8AU1NT";)
    alert tcp any 110 -> any any (msg:"W32/Nimda@mm WORM MAILRECV!!"; flags:PA;
content:"UgEAAI1F6Ild6FCNRfxQU2g/AA8AU1NT";)
```

#### 2．检测 PHPUpload 溢出攻击

PHP 语言为用户提供了上传文件的功能，用户可以使用提供的类进行各类文件、档案的上传功能传送数据给服务器。然而，该类由于没有对上传文件的大小或者类型做严格的判断，在程序的执行过程当中，有可能造成服务器端的缓冲区溢出，从而导致缓冲区溢出攻击。

下面给出防范该溢出攻击的 Snort 检测规则：

```
    alert tcp $EXTERNAL_NET any -> $HOME_NET 80 (msg:"EXPERIMENTAL php content-
disposition"; flags:A+; content:"Content-Disposition\:"; content:"form-data\;";
classtype:web-application-attack; reference:bugtraq,4183; sid:1425; rev:2;)
```

以上规则判断提交给服务器的 HTTP 请求中是否包含“Content-Disposition:”及“form-data;”字符串，如果含有该字符串的话，对于某些没有打补丁的系统来说可能会造成缓冲区溢出攻击。通过添加上述规则，一旦发现客户端有此操作，则 Snort 将会报警。

#### 3．检测 SNMP 口令溢出漏洞

简单网络管理协议（SNMP）是所有基于 TCP/IP 网络上管理不同网络设备的基本协议，比如防火墙、计算机和路由器。现在已经发现，如果攻击者发送怀有恶意信息给 SNMP 的信息接收处理模块，就会引起服务停止（拒绝服务）或缓冲区溢出；或者说通过向运行 SNMP 服务的系统发送一个畸形的管理请求，此时就存在一个缓冲区溢出漏洞，或者造成拒绝服务影响。一旦缓冲区溢出，可以获取部分 SNMP 口令、在本地运行任意的代码以及让攻击者进行任意的操作。因为 SNMP 的程序一般需要系统权限来运行，因此缓冲区溢出攻击可能会造成系统权限被夺取，而形成严重的安全漏洞。

下面给出一条检测 SNMP 口令溢出漏洞的 Snort 规则：

```
    alert udp $EXTERNAL_NET any -> $HOME_NET 161:162 (msg:"EXPERIMENTAL SNMP community string buffer overflow attempt"; content:"|02 01 00 04 82 01 00|"; offset:4; reference:url,www.cert.org/advisories/ CA-2002-03.html; reference:cve,CAN-2002-0012; reference:cve,CAN-2002-0013; classtype:misc-attack; sid:1409; rev:2;)
```

以上规则判断发往 SNMP 服务端口的数据包中是否包含“|02 01 00 04 82 01 00|”二进制串，此串对应 SNMP 操作的分支位置。事实上由于 SNMP 协议的灵活性，对同一分支位置在 SNMP 包里可能有不同的表示，“|02 80 01 80 00 80 04 80 82 80 01 80 00|”就可能表示的是同一分支，更糟的是还有更多的表示方法，攻击者完全可以利用这种协议表示上的灵活性逃过 Snort 的检测，造成漏报。因而要完全解决这个问题，单纯靠搜索特定串是不行的，唯一可行的方法是做协议解码。

4．检测/etc/passwd 文件非法访问

在 Linux 系统中，/etc/passwd 是一个重要的文件，它包含用户名、组成员关系和为用户分配的 shell 等信息。黑客或者不法用户一旦获得了该文件的访问权，就有可能针对该文件进行暴力攻击或者是字典攻击，获得系统的用户和密码，从而获得系统的使用权，对系统将造成极大的威胁。因而，我们需要对/etc/passwd 文件的访问进行检测。

下面给出用于检测/etc/passwd 文件非法访问的 Snort 检测规则：

```
    alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-MISC /etc/passwd";
    flags: A+; content:"/etc/passwd"; nocase; classtype:attempted-recon; sid:1122;
rev:1)
```

上述规则使用字符串匹配算法对包含特征码（“/etc/passwd”字符串）的 HTTP 请求进行检测，一旦发现非法访问，Snort 立刻报警。

## 14.4 主机入侵检测及防御：LIDS

本节内容参考 5.7 节。

## 14.5 分布式入侵检测：SnortCenter

### 14.5.1 分布式入侵检测系统的构成

分布式 Snort 入侵监测系统的主要组成部分包括中央分析服务器、IDS 探针和协同代理。一般情况下，一个企业的网络可以分为外部网络、企业内部网络和 DMZ 区等。入侵监测系统既可以部署在防火墙之外，也可以部署于防火墙之内。部署于防火墙之外，可以获得更多的攻击数据，但是误报警率也高。因此一般情况下，入侵检测系统部署于防火墙之后是比较明智的。

中央分析服务器是整个系统的心脏和灵魂，它一般由一个数据库系统和一个 Web 服务器组成，为系统管理员提供交互式 Web 管理界面。在中央分析服务器上，管理人员可以进行远程 IDS 探针管理、攻击数据查询、实时监视网络攻击及深入分析，如攻击方向的迁移、识别攻击模式等。

MySQL 数据库、Apache 服务器、SnortCenter 和 ACID（Analysis Console for Incident Databases）是一个不错的 Snort 平台系统组合。

SnortCenter 是一个基于 Web 的 Snort 探针和规则管理系统，用于远程修改 Snort 探针的配置、启动、停止探针、编辑、分发 Snort 特征码规则。如果与 cron 和 curl 结合使用，SnortCenter 还可以定时为每个探针进行规则升级，自动从 Snort 官方站点获得最新的特征码规则（http://www.snort.org/dl/signatures/snortrules-stable.tar.gz）。SnortCenter 的下载地址为 http://users.pandora.be/larc/download/。ACID 是一个基于 PHP 的分析引擎，用于查询、管理 Snort 探针产生的数据库。ACID 的下载地址为 http://www.cert.org/kb/acid/。

另外，代理是向中央分析服务器报告攻击信息的软件，是分布式入侵检测系统的一个重要部件。SnortCenter Agent 和普通意义上的协同代理软件有些不同，它只是一个 SnortCenter 进行探针管理的代理工具。通过 SnortCenter Agent，SnortCenter 能够启动、关闭受保护主机上的 Snort 探针；修改探针的配置和使用的特征码规则。SnortCenter Agent 是用 Perl 语言编写的，可以在不同的平台上使用。为了防止登录过程中账户和密码的泄露，它使用 SSL 和 SnortCenter 之间进行通信。Snort 探针是真正进行入侵检测的组件，其源代码可以从 http://www.snort.org 获得。

## 14.5.2 系统安装及部署

### 1. 数据库的安装和设置

在分析服务器上，首先要建立记录 Snort 报警和日志数据库及 SnortCenter、ACID 需要的数据库。在临时目录中解压缩 Snort 的源代码，进入 contrib 目录，建立数据库，并进行相关的配置。配置命令如下：

```
# mysql -u root -p
create database snort;
connect snort
source create_mysql
grant CREATE, INSERT, SELECT, DELETE, UPDATE on snort.* to snort;
grant CREATE, INSERT, SELECT, DELETE, UPDATE on snort.* to snort@localhost;
grant CREATE, INSERT, SELECT, UPDATE on snort.* to acidviewer;
grant CREATE, INSERT, SELECT, UPDATE on snort.* to acidviewer@localhost;
connect mysql
set password for 'snort'@'localhost' = password('yourpassword');
set password for 'snort'@'%' = password('yourpassword');
flush privileges;
```

### 2. 安装 SnortCenter

把最新版的 SnortCenter 源代码解压缩，将目录里面的文件复制到 Apache 的 DocumentRoot 目录（如/usr/local/apache/htdocs）。下载 SnortCenter 需要 adodb（http://php.weblogs.com/adodb/），安装到 SnortCenter 所在的目录，然后建立 SnortCenter 数据库。操作步骤如下：

```
#tar zxvf snortcenter-v0.9.6.tar.gz
#cp snortcenter-v0.9.6/* /usr/local/apache/htdocs/
#tar zxvf adodb250.tgz
#cp -R ./adodb /usr/local/apache/htdocs/
#echo "CREATE DATABASE snortcenter;" | mysql -u root -p
```

完成了基本的安装之后，需要根据系统的情况手工编辑/usr/local/apache/htdocs/config.php 文件，从而结束 SnortCenter 的配置。一般情况下需要修改的参数如表 14-1 所示。

表 14-1　/usr/local/apache/htdocs/config.php 文件需要修改的参数

| 参　数 | 说　明 |
|---|---|
| $DBlib_path | adodb 所在的目录 |
| $curl_path | Curl 命令所在的目录，SnortCenter 使用该命令从官方站点定时获得最新的规则 |
| $Dbtype | 数据库类型 |
| $DB_dbname | 建立的数据库名称 |
| $DB_host | 数据库所在的主机名称 |
| $DB_user | 数据库用户名称 |
| $DB_password | 数据库用户密码 |
| $DB_port | 数据库的服务端口 |
| $hidden_key_num | 随机数串，用于系统安全防护 |

### 3．安装 ACID

ACID 需要 PHP 支持 gd，并对 PHPLot（www.phplot.com）、ADODB 进行支持。新版本的 ACID（0.9.6b22 以后版本）还需要 JpGraph（http://www.aditus.nu/jpgraph/）的支持。ACID 及相关软件安装到指定位置之后，编辑配置文件 acid_conf.php 完成其配置。

另外，ACID 本身缺乏必要的访问控制机制，需要设置 Apache 对访问 ACID 的用户提供验证。

### 4．安装 Snort 探针和 SnortCenter Agent

分别进入需要保护的主机，安装 Snort 和 SnortCenter Agent。SnortCenter Agent 需要 SSL 的支持，所需模块可以从 http://www.cpan.org 得到。安装命令如下：

```
#./configure -with-mysql
# make
# make install
# tar -zxvf Net_SSLeay.pm.1.21.tar.gz
# cd Net_ SSLeay.pm.1.21
# perl Makefile.PL
# make install
# mkdir /opt/snortagent/
```

```
# cp snortcenter-agent-v0.1.6.tar.gz /opt/snortagent
# cd /opt/snortagent
# tar -zxvf snortcenter-agent-v0.1.6.tar.gz
# cd sensor
# ./setup.sh
```

# 第四篇

# 企业 Linux 安全监控

## 师夷长技以制夷

“师夷长技以制夷”是指通过学习西方的先进军事技术寻求御侮强国之道。后来指：学习西方的先进技术来抵制西方，也就是“以彼之道，还之彼身”，是魏源在他的《海国图志》中提出来的。

在企业 Linux 安全监控工作中，此方法主要是指无论是管理员还是信息安全工作人员都需要站在黑客的角度来思考，从各个技术细节来对系统和网络可能遭受的攻击和外部威胁进行安全监控，这样才能在实际的运维工作中处于有利地位，保证企业信息安全。

# 第 15 章

# 管中窥豹：企业 Linux 系统及性能监控

## 本章导读

500 强企业信息安全工作除了本书前面部分讲述的安全原理、安全架构和实施之外，很大一部分工作就在于安全运维。运维是安全必不可少的必修课和必要阶段。具体来讲，前面部署的安全架构和服务器需要从系统层面、性能层面来进行监控，从而发现系统及其承载的业务在运行中出现的问题，并尽快地进行解决，这是安全运维保障系统和服务可用性的重要工作内容。

基于此，本章将详细介绍企业 Linux 系统及性能监控的常用工具和方法，并通过大量的实例来揭示 500 强企业是如何进行此项安全运维工作的。

# 15.1 常用的性能监测工具

Linux 系统下，大多数的性能监测工具保存在/proc 目录下。这里我们将 Linux AS 和 SUSE Linux Enterprise Server 中的命令行及图形方式下的性能监测工具做概括性的介绍。这些工具有些在系统工具盘里，有些可以从网上下载。sar、iostat、和 pstat 这三个工具在 distributionCD 里，也可以从网上下载，网址是 http://perso.wanadoo.fr/sebastien.godard/。Linux 性能监测工具如表 15-1 所示。

表 15-1 Linux 性能监测工具

| 工　具 | 说　明 |
|---|---|
| uptime | 系统平均负载 |
| dmesg | 硬件/系统信息 |
| top | 处理器活动情况 |
| iostat | CPU 平均负载和磁盘情况 |
| vmstat | 系统情况 |
| sar | 收集和报告系统情况 |
| KDE System Guard | 实时的系统情况报告和图表展现 |
| free | 内存使用情况 |
| Traffic-vis | 网络监控（SUSE Linux 企业级服务器专用） |
| pmap | 处理器内存使用情况 |
| strace | 系统程序情况 |
| ulimit | 系统限制 |
| mpstat | 多处理器使用情况 |

## 15.1.1 uptime

uptime 命令用于查看服务器运行了多长时间以及有多少个用户登录，快速获取服务器的负荷情况。

uptime 的输出包含一项内容 load average，显示了最近 1-、5-、15 分钟的负荷情况。其值代表等待 CPU 处理的进程数，如果 CPU 没有时间处理这些进程，load average 值会升高；反之则会降低。load average 的最佳值是 1，说明每个进程都可以马上处理并且没有 CPU cycles 被丢失。对于单 CPU 的机器，1 或者 2 是可以接受的值；对于多路 CPU 的机器，load average 值可能在 8～10 之间。也可以使用 uptime 命令来判断网络性能。例如，某个网络应用性能很低，通过运行 uptime 查看服务器的负荷是否很高，如果不是，那么问题应该是网络方面造成的。

下面是 uptime 命令的输出样式：

```
15:09:50 up 98 days,  5:16, 16 users,  load average: 0.77, 1.06, 1.08
```

## 15.1.2 dmesg

dmesg 命令主要用来显示内核信息。使用 dmesg 可以有效诊断机器硬件故障或者添加硬件出现的问题。

使用 dmesg 命令可以确定您的服务器安装了哪些硬件。每次系统重启，系统都会检查所有硬件并将信息记录下来。执行/bin/dmesg 命令可以查看该记录。

下面是 dmesg 命令的输出样式：

```
# dmesg |more 30
6593]: /dev/rtc enable interrupt failed: -515
/dev/vmmon[26593]: /dev/rtc enable interrupt failed: -515
/dev/vmmon[26593]: /dev/rtc enable interrupt failed: -515
/dev/vmmon[26593]: /dev/rtc enable interrupt failed: -515
HostIF_MapUserMem: p = 0x0000000000e32804, offset = 0x0000000000000804,
numPagesNeeded = 1, handleSize = 24, mappedAddr = 0xffff810838010000
/dev/vmmon[26593]: /dev/rtc enable interrupt failed: -515
/dev/vmmon[26593]: /dev/rtc enable interrupt failed: -515
/dev/vmmon[26593]: /dev/rtc enable interrupt failed: -515
HostIF_UnmapUserMem: numPages = 1, addr = 0xffff810838010000
HostIF_UnmapUserMem: numPages = 1, addr = 0xffff810838010000
HostIF_UnmapUserMem: numPages = 1, addr = 0xffff810838010000
/dev/vmmon[27544]: PTSC: initialized at 2933400000 Hz using TSC, TSCs are
synchronized.
/dev/vmnet: open called by PID 27567 (vmx-vcpu-0)
/dev/vmnet: port on hub 8 successfully opened
/dev/vmnet: open called by PID 27567 (vmx-vcpu-0)
/dev/vmnet: port on hub 8 successfully opened
HostIF_MapUserMem: p = 0x0000000000e327f8, offset = 0x00000000000007f8,
numPagesNeeded = 1, handleSize = 24, mappedAddr = 0xffff81030d1e9000
HostIF_MapUserMem: p = 0x0000000000e3281c, offset = 0x000000000000081c,
numPagesNeeded = 1, handleSize = 24, mappedAddr = 0xffff81030d1e9000
/dev/vmmon[27567]: /dev/rtc enable interrupt failed: -515
/dev/vmmon[27567]: /dev/rtc enable interrupt failed: -515
HostIF_MapUserMem: p = 0x0000000000e32804, offset = 0x0000000000000804,
numPagesNeeded = 1, handleSize = 24, mappedAddr = 0xffff81030d1e9000
/dev/vmmon[27567]: /dev/rtc enable interrupt failed: -515
/dev/vmmon[27567]: /dev/rtc enable interrupt failed: -515
/dev/vmnet: open called by PID 28168 (vmx-vcpu-0)
/dev/vmnet: port on hub 8 successfully opened
/dev/vmnet: open called by PID 28168 (vmx-vcpu-0)
/dev/vmnet: port on hub 8 successfully opened
/dev/vmmon[28168]: /dev/rtc enable interrupt failed: -515
```

### 15.1.3 top

top 命令显示处理器的活动状况。默认情况下，显示占用 CPU 最多的任务，并且每隔 5 秒做一次刷新。

Process priority 的数值决定了 CPU 处理进程的顺序。Liunx 内核会根据需要调整该数值的大小。nicevalue 局限于 priority。priority 的值不能低于 nice value（nicevalue 值越低，优先级越高）。不可以直接修改 Process priority 的值，但是可以通过调整 nicelevel 的值来间接地改变 Process priority 值，然而这一方法并不是所有时候都可用。如果某个进程运行异常缓慢，可以通过降低 nicelevel 为该进程分配更多的 CPU。

Linux 支持的 nicelevels 由 19（优先级低）～-20（优先级高），默认值为 0。执行/bin/ps 命令可以查看到当前进程的情况。

### 15.1.4 iostat

iostat 由 Red Hat Enterprise Linux AS 发布。同时 iostat 也是 Sysstat 的一部分，可以下载到，网址是 http://perso.wanadoo.fr/sebastien.godard/。

执行 iostat 命令可以查看从系统启动之后的 CPU 平均时间，类似于 uptime。除此之外，iostat 还能创建一个服务器磁盘子系统的活动报告。该报告包含两部分：CPU 使用情况和磁盘使用情况。

下面是 iostat 命令的输出样式：

```
# iostat
Linux 2.6.18-194.el5      2012/08/15

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           1.28    0.03    1.17    0.38    0.00   97.14

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
sda               2.19         5.46        46.86   46353466  397661300
sda1              0.00         0.00         0.00       4138        784
sda2              2.18         5.43        46.83   46119346  397386104
sda3              0.00         0.00         0.00       1880          0
sda4              0.00         0.00         0.00         10          0
sda5              0.00         0.03         0.03     227660     274412
sdb              15.87        53.05       266.02  450236707 2257534121
sdb1             15.87        53.05       266.02  450235851 2257534121
```

CPU 占用情况包括四部分内容。

- %user：显示 user level（applications）时，CPU 的占用情况。
- %nice：显示 user level 在 nice priority 时，CPU 的占用情况。
- %sys：显示 system level（kernel）时，CPU 的占用情况。

- %idle：显示 CPU 空闲时间所占比例。

磁盘使用报告分成以下几个部分。

- Device：该设备的名字。
- tps：该设备每秒 I/O 传输的次数。多个 I/O 请求可以组合为一个，每个 I/O 请求传输的字节数不同，因此可以将多个 I/O 请求合并为一个。
- Blk_read/s、Blk_wrtn/s：表示从该设备每秒读写的数据块数量。块的大小可以不同，如 1024、2048 或 4048B，这取决于 partition 的大小。

例如，执行下列命令获得设备/dev/sda1 的数据块大小：

```
#dumpe2fs -h /dev/sda1 |grep -F "Block size"
```

输出结果如下：

```
dumpe2fs 1.34 (25-Jul-2003)
Block size: 1024
Blk_read, Blk_wrtn: 指示自从系统启动之后数据块读/写的合计数。
```

### 15.1.5 vmstat

vmstat 提供了 processes、memory、paging、block I/O、traps 和 CPU 的活动状况。

下面是 vmstat 命令的输出样式：

```
procs ---------memory------- ---swap-- -----io---- --system-- -----cpu------
r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
3  0      0 10311240 553192 15478076    0    0     1     7    0    0  1  1 97  0  0
```

各输出列的含义如下。

- procss：
  - -r：等待 runtime 的进程数。
  - -b：在不可打断的休眠状态下的进程数。
- memory：
  - -swpd：虚拟内存使用量（KB）。
  - -free：闲置内存使用量（KB）。
  - -buff：被当作 buffer 使用的内存量（KB）。
- swap：
  - -si：swap 到磁盘的内存量（KB/s）。
  - -so：从磁盘 swap 出去的内存量（KB/s）。
- io：
  - -bi：发送给块设备的数据块数目（blocks/s）。
  - -bo：从块设备接收的数据块数目（blocks/s）。
- system：
  - -in：每秒产生的中断数目（包括时钟中断）。
  - -cs：每秒产生的上下文切换数目。
- cpu（these are percentages of total CPU time）：

- -us：在非内核运行消耗的时间。
- -sy：在内核运行消耗的时间。
- -id：空闲时间。
- -wa：I/O 等待时间。

### 15.1.6 sar

sar 是 Red Hat Enterprise Linux AS 发行的一个工具，同时也是 Sysstat 工具集的命令之一。sar 用于收集、报告或者保存系统活动信息。sar 由三个应用组成：sar 显示数据、sar1 和 sar2 用于收集和保存数据。

使用 sar1 和 sar2，系统能够配置自动抓取信息和日志，以备分析使用。配置举例：在/etc/crontab 中添加以下几行内容：

```
# 8am-7pm activity reports every 10 minutes during weekdays.
*/10 8-18 * * 1-5 /usr/lib/sa/sa1 600 6 &
# 7pm-8am activity reports every an hour during weekdays.
0 19-7 * * 1-5 /usr/lib/sa/sa1 &
# Activity reports every an hour on Saturday and Sunday.
0 * * * 0,6 /usr/lib/sa/sa1 &
# Daily summary prepared at 19:05
5 19 * * * /usr/lib/sa/sa2 -A &
```

同样的，也可以在命令行方式下使用 sar 运行实时报告：

```
# sar -u 1 10
Linux 2.6.18-194.el5      08/15/12

15:19:24       CPU    %user    %nice   %system   %iowait    %steal    %idle
15:19:25       all     1.79     0.00      0.29      0.00      0.00    97.92
15:19:26       all     2.58     0.21      0.25      0.17      0.00    96.80
15:19:27       all     1.71     0.00      0.21      0.00      0.00    98.08
15:19:28       all     1.58     0.00      0.17      0.04      0.00    98.21
15:19:29       all     1.79     0.00      0.25      0.00      0.00    97.96
15:19:30       all     1.83     0.00      0.12      0.00      0.00    98.04
15:19:31       all     1.87     0.00      0.25      0.12      0.00    97.75
15:19:32       all     1.83     0.00      0.17      0.00      0.00    98.00
15:19:33       all     1.92     0.00      0.08      0.00      0.00    98.00
15:19:34       all     1.87     0.00      0.21      0.08      0.00    97.83
Average:       all     1.88     0.02      0.20      0.04      0.00    97.86
```

从收集的信息中，可以得到详细的 CPU 使用情况（%user、%nice、%system、%idle）、内存页面调度、网络 I/O、进程活动、块设备活动，以及每秒产生的中断数目。

### 15.1.7 KDE System Guard

KDE System Guard（KSysguard）指 KDE 任务管理和性能监视。监视本地及远程客户端/服务器架构体系中的主机。

如图 15-1 所示，使用传感器获取显示的信息。传感器可以返回简单的数值或者复杂的表格信息。

对于每一种类型的信息，提供了一个或者更多显示，并以工作表的形式独立保存。

每个传感器监视一个部件。所有显示的传感器均可以用鼠标拖曳。有以下三个选择。

- 可以删除和替换某个传感器。
- 可以编辑修改行数和列数。
- 可以建立新的工作表并选择所需的传感器。

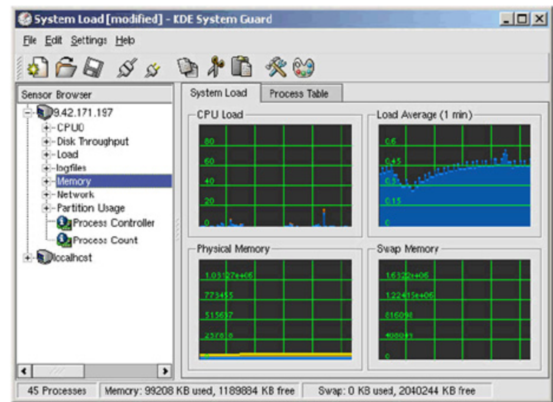



图 15-1　KDE System Guard 示意图

## 15.1.8　free

/bin/free 命令显示所有空闲的和使用的内存数量，包括 swap，同时也包含内核使用的缓存。

```
# free
total       used       free     shared    buffers     cached
Mem:      32933832   22640856   10292976          0     553468   15480308
-/+ buffers/cache:    6607080   26326752
Swap:            0          0          0
```

## 15.1.9　Traffic-vis

Traffic-vis 是一套测定哪些主机在 IP 网进行通信、通信的目标主机以及传输的数据量，并输出纯文本、HTML 或者 GIF 格式的报告。值得注意的是：Traffic-vis 仅仅适用于 SUSE Linux Enterprise Server。

可以使用以下命令来收集 eth0 的信息：

```
#traffic-collector -i eth0 -s /root/output_traffic-collector
```

可以使用 killall 命令来控制该进程。如果要将报告写入磁盘，使用以下命令：

```
#killall -SIGUSR1 traffic-collector
```

要停止对信息的收集，执行以下命令：

```
killall -SIGTERM traffic-collector
```

尤其值得注意的是：不要忘记执行最后一条命令，否则会因为内存占用而影响性能。

可以根据 packet、bytes、TCP 连接数对输出进行排序，根据每项的总数或者收/发的数量进行。例如根据主机上 packets 的收/发数量排序，执行以下命令：

```
#traffic-sort -i output_traffic-collector -o output_traffic-sort -Hp
```

如要生成 HTML 格式的报告，显示传输的字节数、packets 的记录、全部 TCP 连接请求和网络中每台服务器的信息，请运行以下命令：

```
#traffic-tohtml -i output_traffic-sort -o output_traffic-tohtml.html
```

使用管道，可以只需执行一条命令来产生报告。如生成 HTML 的报告，执行以下命令：

```
#cat output_traffic-collector | traffic-sort -Hp | traffic-tohtml-o output_traffic-tohtml.html
```

如要生成 GIF 文件，执行以下命令：

```
#cat output_traffic-collector | traffic-sort -Hp | traffic-togif-o output_traffic-togif.gif -x 600 -y 600
```

### 15.1.10 pmap

pmap 可以报告某个或多个进程的内存使用情况。使用 pmap 判断主机中哪个进程因占用过多内存导致内存瓶颈。以下为在 Red Hat Enterprise Server 下 pmap 命令的执行结果：

```
# pmap -x 30536
30536:   /usr/lib64/firefox-3.0.18/firefox -UILocale zh-CN
Address           Kbytes     RSS   Dirty Mode   Mapping
0000000000400000      40      32       0 r-x--  firefox
000000000060a000       4       4       4 rw---  firefox
0000000015f7f000   66176   66120   66120 rw---    [ anon ]
0000000040100000       4       0       0 -----    [ anon ]
0000000040101000   10240       8       8 rw---    [ anon ]
00000000415e0000       4       0       0 -----    [ anon ]
00000000415e1000   10240       8       8 rw---    [ anon ]
0000000041fe1000       4       0       0 -----    [ anon ]
0000000041fe2000   10240       8       8 rw---    [ anon ]
00000000429e2000       4       0       0 -----    [ anon ]
00000000429e3000   10240      16      16 rw---    [ anon ]
00000000433e3000       4       0       0 -----    [ anon ]
00000000433e4000   10240       8       8 rw---    [ anon ]
0000000043de4000       4       0       0 -----    [ anon ]
0000000043de5000   10240      12      12 rw---    [ anon ]
00000000447e5000       4       0       0 -----    [ anon ]
00000000447e6000   10240      20      20 rw---    [ anon ]
00000000451e6000       4       0       0 -----    [ anon ]
00000000451e7000   10240      16      16 rw---    [ anon ]
0000003311200000      68      52       0 r-x--  libresolv-2.5.so
0000003311211000    2048       0       0 -----  libresolv-2.5.so
0000003311411000       4       4       4 r----  libresolv-2.5.so
0000003311412000       4       4       4 rw---  libresolv-2.5.so
```

```
0000003311413000      8       4       0 rw---    [ anon ]
0000003311a00000    176      40       0 r-x--  libgssapi_krb5.so.2.2
0000003311a2c000   2048       0       0 -----  libgssapi_krb5.so.2.2
0000003311c2c000      8       8       8 rw---  libgssapi_krb5.so.2.2
0000003312a00000    144      28       0 r-x--  libk5crypto.so.3.1
0000003312a24000   2044       0       0 -----  libk5crypto.so.3.1
0000003312c23000      8       8       8 rw---  libk5crypto.so.3.1
0000003314600000    280      52       0 r-x--  libssl.so.0.9.8e
0000003314646000   2048       0       0 -----  libssl.so.0.9.8e
0000003314846000     24      12      12 rw---  libssl.so.0.9.8e
0000003315800000    580      88       0 r-x--  libkrb5.so.3.3
0000003315891000   2048       0       0 -----  libkrb5.so.3.3
0000003315a91000     16      16      12 rw---  libkrb5.so.3.3
0000003315c00000     32      16       0 r-x--  libkrb5support.so.0.1
0000003315c08000   2044       0       0 -----  libkrb5support.so.0.1
0000003315e07000      4       4       4 rw---  libkrb5support.so.0.1
0000003d54a00000    112     104       0 r-x--  ld-2.5.so
0000003d54c1b000      4       4       4 r----  ld-2.5.so
```

下面显示了 httpd 进程所占用的内存：

```
# pmap -x 7811
7811:   /usr/sbin/httpd
Address           Kbytes     RSS   Dirty Mode   Mapping
00002b54c58fa000    308     192       0 r-x--  httpd
00002b54c5947000    100       0       0 rw-s-  zero (deleted)
00002b54c5b47000     16      16      16 rw---  httpd
00002b54c5b4b000     12       8       8 rw---    [ anon ]
00002b54c5b4e000    112      40       0 r-x--  ld-2.5.so
00002b54c5b6a000      4       4       4 rw---    [ anon ]
00002b54c5b94000      4       4       4 rw---    [ anon ]
00002b54c5d69000      4       4       4 r----  ld-2.5.so
00002b54c5d6a000      4       4       4 rw---  ld-2.5.so
00002b54c5d6b000    520       4       0 r-x--  libm-2.5.so
00002b54c5ded000   2044       0       0 -----  libm-2.5.so
00002b54c5fec000      4       4       4 r----  libm-2.5.so
00002b54c5fed000      4       4       4 rw---  libm-2.5.so
00002b54c5fee000    108      28       0 r-x--  libpcre.so.0.0.1
```

```
00002b54c6009000   2048      0      0 -----  libpcre.so.0.0.1
00002b54c6209000      4      4      4 rw---  libpcre.so.0.0.1
00002b54c620a000     84      4      0 r-x--  libselinux.so.1
00002b54c621f000   2048      0      0 -----  libselinux.so.1
00002b54c641f000      8      8      8 rw---  libselinux.so.1
---------------- ------ ------ ------
total kB         239836   6944   5708
```

## 15.1.11 strace

strace 截取和记录系统进程调用，以及进程收到的信号。是一个非常有效的检测、指导和调试工具。系统管理员可以通过该命令容易地解决程序问题。

使用该命令需要指明进程的 ID（PID），例如：

```
# strace -p 27215
Process 27215 attached - interrupt to quit
select(256, [0 1 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27], NULL, NULL, NULL) = 1 (in [7])
select(8, [7], NULL, NULL, {0, 0})     = 1 (in [7], left {0, 0})
read(7, "\4\0@\0\0\0\0l", 8192)        = 8
select(8, [7], NULL, NULL, {0, 0})     = 0 (Timeout)
setitimer(ITIMER_REAL, {it_interval={0, 20000}, it_value={0, 20000}}, NULL) = 0
writev(24, [{"\3.\16F\245\3404):\0\0\0|\0\0\2\0\0\0\0\33\1\351\0\34\1\271\0\0\0\1\0", 32}], 1) = 32
select(256, [0 1 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27], NULL, NULL, NULL) = ? ERESTARTNOHAND (To be restarted)
--- SIGALRM (Alarm clock) @ 0 (0) ---
setitimer(ITIMER_REAL, {it_interval={0, 0}, it_value={0, 0}}, NULL) = 0
rt_sigreturn(0)                        = -1 EINTR (Interrupted system call)
select(256, [0 1 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27], NULL, NULL, NULL) = 1 (in [25])
setitimer(ITIMER_REAL, {it_interval={0, 20000}, it_value={0, 20000}}, NULL) = 0
read(25, "&\1\2\0:\0\0\0", 4096)       = 8
read(25, 0x1157da60, 4096)      = -1 EAGAIN (Resource temporarily unavailable)
writev(25,
[{"\1\1\343\306\0\0\0\0:\0\0\0\306T\246\0\33\1\351\0\33\1\351\0\0\0\242\0\0\0\0\0", 32}], 1) = 32
select(256, [0 1 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27], NULL, NULL, NULL) = ? ERESTARTNOHAND (To be restarted)
```

```
--- SIGALRM (Alarm clock) @ 0 (0) ---
setitimer(ITIMER_REAL, {it_interval={0, 0}, it_value={0, 0}}, NULL) = 0
rt_sigreturn(0)                     = -1 EINTR (Interrupted system call)
select(256, [0 1 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
25 26 27], NULL, NULL, NULL) = 1 (in [7])
select(8, [7], NULL, NULL, {0, 0})     = 1 (in [7], left {0, 0})
read(7, "\4\1@\0\0\0\377\10", 8192)    = 8
select(8, [7], NULL, NULL, {0, 0})     = 0 (Timeout)
setitimer(ITIMER_REAL, {it_interval={0, 20000}, it_value={0, 20000}}, NULL) = 0
```

### 15.1.12 ulimit

ulimit 内置在 bash shell 中，用来提供对 shell 和进程可用资源的控制，使用选项-a 列出可以设置的所有参数：

```
# ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 278528
max locked memory       (kbytes, -l) 32
max memory size         (kbytes, -m) unlimited
open files                      (-n) 1024
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) 10240
cpu time               (seconds, -t) unlimited
max user processes              (-u) 278528
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
```

-H 和-S 选项指明所给资源的软、硬限制。如果超过了软限制，系统管理员会收到警告信息。硬限制指在用户收到超过文件句柄限制的错误信息之前，可以达到的最大值。

例如：可以设置对文件句柄的硬限制：#ulimit -Hn 4096，同样，可以设置对文件句柄的软限制：#ulimit -Sn 1024。如果要查看软、硬限制值，执行以下命令：

```
#ulimit -Hn
```

```
#ulimit -Sn
```

例如限制 Oracle 用户，在/etc/security/limits.conf 输入以下行：

```
soft nofile 4096
hard nofile 10240
```

对于 Red Hat Enterprise Linux AS，确定文件/etc/pam.d/system-auth 包含以下行：

```
session required /lib/security/#ISA/pam_limits.so
```

对于 SUSE Linux Enterprise Server，确定文件/etc/pam.d/login 和/etc/pam.d/sshd 包含以下行：

```
session required pam_limits.so
```

这一行使这些限制生效。

### 15.1.13 mpstat

mpstat 是 Sysstat 工具集的一部分。mpstat 用于报告多路 CPU 主机每颗 CPU 的活动情况，以及整个主机的 CPU 情况。

例如，下面的命令可以隔 3 秒报告一次处理器的活动情况，执行 4 次：

```
# mpstat 3 4
Linux 2.6.18-194.el5      08/15/12

15:29:06 CPU %user %nice %sys %iowait %irq %soft %steal %idle intr/s
15:29:09 all 1.97 0.06 0.12 0.12 0.01 0.01 0.00 97.70 1835.22
15:29:12 all 1.49 0.00 0.06 0.11 0.00 0.01 0.00 98.33 1118.67
15:29:15 all 1.58 0.00 0.08 0.03 0.00 0.00 0.00 98.31 1089.30
15:29:18 all 2.14 0.06 0.11 0.15 0.00 0.00 0.00 97.54 1110.67
Average: all 1.79 0.03 0.09 0.10 0.00 0.01 0.00 97.97 1289.08
```

以下命令每隔 1 秒显示一次多路 CPU 主机的处理器活动情况，执行 2 次：

```
# mpstat -P ALL 1 2
Linux 2.6.18-194.el5      08/15/12

15:30:18 CPU %user %nice %sys %iowait  %irq %soft %steal %idle intr/s
15:30:19 all 1.95 0.00 0.12 0.00 0.00 0.00 0.00 97.92 1056.00
15:30:19 0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 1002.00
15:30:19 1 2.97 0.00 0.00 0.00 0.00 0.00 0.00 97.03 1.00
15:30:19 2 2.97 0.00 0.00 0.00 0.00 0.00 0.00 97.03 18.00
15:30:19 3 3.00 0.00 0.00 0.00 0.00 0.00 0.00 97.00 1.00
15:30:19 4 2.00 0.00 0.00 0.00 0.00 0.00 0.00 98.00 0.00
15:30:19 5 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 1.00
15:30:19 6 1.00 0.00 0.00 0.00 0.00 0.00 0.00 99.00 0.00
```

```
15:30:19 7 1.00 0.00 0.00 0.00 0.00 0.00 0.00 99.00 0.00
15:30:19 8 2.97 0.00 0.99 0.00 0.00 0.00 0.00 96.04 0.00
15:30:19 9 8.00 0.00 0.00 0.00 0.00 0.00 0.00 92.00 0.00
15:30:19 10 2.00 0.00 0.00 0.00 0.00 0.00 0.00 98.00 0.00
15:30:19 11 2.00 0.00 0.00 0.00 0.00 0.00 0.00 98.00 0.00
15:30:19 12 1.01 0.00 0.00 0.00 0.00 0.00 0.00 98.99 0.00
15:30:19 13 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00
15:30:19 14 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 4.00
15:30:19 15 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 9.00
15:30:19 16 1.98 0.00 0.00 0.00 0.00 0.00 0.00 98.02 0.00
15:30:19 17 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 18.00
15:30:19 18 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00
15:30:19 19 2.00 0.00 0.00 0.00 0.00 0.00 0.00 98.00 0.00
15:30:19 20 2.00 0.00 0.00 0.00 0.00 0.00 0.00 98.00 0.00
15:30:19 21 5.94 0.00 0.99 0.00 0.00 0.00 0.00 93.07 0.00
15:30:19 22 7.00 0.00 0.00 0.00 0.00 0.00 0.00 93.00 0.00
15:30:19 23 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00

15:30:19 CPU %user %nice %sys %iowait %irq %soft %steal %idle intr/s
15:30:20 all 2.12 0.17 0.38 0.00 0.00 0.00 0.00 97.33 1494.00
15:30:20 0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 1001.00
15:30:20 1 2.00 0.00 0.00 0.00 0.00 0.00 0.00 98.00 2.00
15:30:20 2 2.02 0.00 0.00 0.00 0.00 0.00 0.00 97.98 4.00
15:30:20 3 1.98 0.00 0.99 0.00 0.00 0.00 0.00 97.03 0.00
15:30:20 4 4.00 0.00 0.00 0.00 0.00 0.00 0.00 96.00 390.00
15:30:20 5 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00
15:30:20 6 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00
15:30:20 7 1.00 4.00 0.00 0.00 0.00 0.00 0.00 95.00 0.00
15:30:20 8 2.00 0.00 0.00 0.00 0.00 0.00 0.00 98.00 0.00
15:30:20 9 2.02 0.00 0.00 0.00 0.00 0.00 0.00 97.98 0.00
15:30:20 10 2.00 0.00 2.00 0.00 0.00 0.00 0.00 96.00 0.00
15:30:20 11 1.98  0.00 0.00 0.00 0.00 0.00 0.00 98.02 0.00
15:30:20 12 13.73 0.00 0.98 0.00 0.00 0.00 0.00 85.29 0.00
15:30:20 13 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00
15:30:20 14 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00
15:30:20 15 1.00 0.00 1.00 0.00 0.00 0.00 0.00 98.00 11.00
```

```
15:30:20 16 1.01 0.00 0.00 0.00 0.00 0.00 0.00 98.99 0.00
15:30:20 17 0.00 0.00 1.00 0.00 0.00 0.00 0.00 99.00 86.00
15:30:20 18 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00
15:30:20 19 2.97 0.00 0.00 0.00 0.00 0.00 0.00 97.03 0.00
15:30:20 20 2.02 0.00 0.00 0.00 0.00 0.00 0.00 97.98 0.00
15:30:20 21 7.92 0.00 2.97 0.00 0.00 0.00 0.00 89.11 0.00
15:30:20 22 2.00 0.00 0.00 0.00 0.00 0.00 0.00 98.00 0.00
15:30:20 23 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00

Average:CPU %user %nice %sys %iowait %irq %soft %steal %idle intr/s
Average:all 2.04 0.08 0.25 0.00 0.00 0.00 0.00 97.63 1275.00
Average:0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 1001.50
Average:1 2.49 0.00 0.00 0.00 0.00 0.00 0.00 97.51 1.50
Average:2 2.50 0.00 0.00 0.00 0.00 0.00 0.00 97.50 11.00
Average:3 2.49 0.00 0.50 0.00 0.00 0.00 0.00 97.01 0.50
Average:4 3.00 0.00 0.00 0.00 0.00 0.00 0.00 97.00 195.00
Average:5 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.50
Average:6 0.50 0.00 0.00 0.00 0.00 0.00 0.00 99.50 0.00
Average:7 1.00 2.00 0.00 0.00 0.00 0.00 0.00 97.00 0.00
Average:8 2.49 0.00 0.50 0.00 0.00 0.00 0.00 97.01 0.00
Average:9 5.03 0.00 0.00 0.00 0.00 0.00 0.00 94.97 0.00
Average:10 2.00 0.00 1.00 0.00 0.00 0.00 0.00 97.00 0.00
Average:11 1.99 0.00 0.00 0.00 0.00 0.00 0.00 98.01 0.00
Average:12 7.46 0.00 0.50 0.00 0.00 0.00 0.00 92.04 0.00
Average:13 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00
Average:14 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 2.00
Average:15 0.50 0.00 0.50 0.00 0.00 0.00 0.00 99.00 10.00
Average:16 1.50 0.00 0.00 0.00 0.00 0.00 0.00 98.50 0.00
Average:17 0.00 0.00 0.50 0.00 0.00 0.00 0.00 99.50 52.00
Average:18 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00
Average:19 2.49 0.00 0.00 0.00 0.00 0.00 0.00 97.51 0.00
Average:20 2.01 0.00 0.00 0.00 0.00 0.00 0.00 97.99 0.00
Average:21 6.93 0.00 1.98 0.00 0.00 0.00 0.00 91.09 0.00
Average:22 4.50 0.00 0.00 0.00 0.00 0.00 0.00 95.50 0.00
Average:23 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00 0.00
```

## 15.2 CPU 监控详解

一般来说，CPU 监控主要对其利用率进行实时监控，CPU 利用率主要依赖于是什么资源在试图存取。内核调度器将负责调度两种资源种类：线程（单一或者多路）和中断。调度器会定义不同资源的不同优先权。以下列表按优先级从高到低排列。

- Interrupts（中断）。设备通知内核，它们完成一次数据处理的过程。例如，当一块网卡设备传送网络数据包或者一块硬件提供了一次 IO 请求。
- KernelProcesses（内核处理过程）。所有内核处理过程就是控制优先级别。
- User Processes（用户进程）。所有软件程序都运行在这个 user space。该块在内核调度机制中处于低优先级。

从上面我们可以看出内核是怎样管理不同资源的，还有几个关键内容需要介绍。以下部分就将介绍 context（上下文切换）、run queues（运行队列）以及 utilization（利用率）。

### 15.2.1 上下文切换

多数现代处理器都能够运行一个进程（单一线程）或者线程。多路超线程处理器有能力运行多个线程。然而，Linux 内核还是把每个处理器核心的双核心芯片作为独立的处理器。比如，以 Linux 内核的系统在一个双核心处理器上，是报告显示为两个独立的处理器。

一个标准的 Linux 内核可以运行 50～50 000 个处理线程。在只有一个 CPU 时，内核将调度并均衡每个进程线程，每个线程都分配一个在处理器中被开销的时间额度。一个线程要么就是获得时间额度或已抢先获得一些具有较高优先级（比如硬件中断），其中较高优先级的线程将从区域重新放置回处理器的队列中。这种线程的转换关系就是我们提到的上下文切换。

每次内核的上下文切换，资源被用于关闭在 CPU 寄存器中的线程和放置在队列中。系统中的上下文切换越多，在处理器的调度管理下，内核将得到更多的工作。

### 15.2.2 运行队列

每个 CPU 都维护一个线程的运行队列。理论上，调度器应该不断地运行和执行线程。进程线程不是在 sleep 状态中（阻塞中和等待 IO 中）就是在可运行状态中。如果 CPU 子系统处于高负荷状态下，那就意味着内核调度将无法及时响应系统请求。导致的结果是可运行状态进程拥塞在运行队列里。当运行队列越来越巨大时，进程线程将花费更多的时间获取被执行。

比较流行的术语就是“load”，它提供当前运行队列的详细状态。系统 load 就是指在 CPU 队列中有多少数目的线程，以及其中当前有多少进程线程数目被执行的组合。如果一个双核系统执行了两个线程，还有 4 个在运行队列中，则 load 应该为 6。top 程序里显示的 load averages 是指 1、5、15 分钟以内的 load 情况。

### 15.2.3 CPU 利用率

CPU 利用率定义 CPU 使用的百分比。评估系统最重要的一个度量方式就是 CPU 的利用率，多数性能监控工具关于 CPU 利用率的分类有以下几种。

- User Time（用户进程时间）：在 user space 中被执行进程在 CPU 中开销的时间百分比。

- System Time（内核线程以及中断时间）：在 kernel space 中线程和中断在 CPU 中开销的时间百分比。
- Wait IO（IO 请求等待时间）：所有进程线程被阻塞等待完成一次 IO 请求所占 CPU 开销 idle 的时间百分比。
- Idle（空闲）：一个完整空闲状态的进程在 CPU 处理器中开销的时间百分比。

### 15.2.4　使用 vmstat 工具进行监控

vmstat 工具提供了一种低开销的系统性能观察方式。因为 vmstat 本身就是低开销工具，在非常高负荷的服务器上，你需要查看并监控系统的健康情况，在控制窗口还能够使用 vmstat 输出结果。该工具运行在两种模式（verage 和 sample 模式）下。sample 模式通过指定间隔时间测量状态值，该模式对于理解在持续负荷下的性能表现很有帮助。下面给出的是 vmstat 运行 1 秒间隔的示例：

```
# vmstat 1
procs ———-memory——-- —swap——-io—- -system——-cpu—-
r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs us  sy id  wa
0  0 104300  16800  95328  72200    0    0     5    26    7    14  4   1 95   0
0  0 104300  16800  95328  72200    0    0     0    24 1021    64  1   1 98   0
0  0 104300  16800  95328  72200    0    0     0     0 1009    59  1   1 98   0
```

其中，各字段的含义如下。

- r：当前运行队列中线程的数目。代表线程处于可运行状态，但 CPU 还未能执行。
- b：当前进程阻塞并等待 IO 请求完成的数目。
- in：当前中断被处理的数目。
- cs：发生上下文切换的数目。
- us：CPU 利用率的百分比。
- sys：内核和中断利用率的百分比。
- wa：所有可运行状态线程被阻塞在等待 IO 请求的百分比。
- id：CPU 空闲时间的百分比。

下面给出两个具体的例子来说明如何进行判别。

**例子 1**：在下面的例子中，系统被充分利用：

```
# vmstat 1
procs                  memory      swap          io     system         cpu
r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs us sy wa id
3  0 206564  15092  80336 176080    0    0     0     0  718    26 81 19  0  0
2  0 206564  14772  80336 176120    0    0     0     0  758    23 96  4  0  0
1  0 206564  14208  80336 176136    0    0     0     0  820    20 96  4  0  0
1  0 206956  13884  79180 175964    0  412     0  2680 1008    80 93  7  0  0
2  0 207348  14448  78800 175576    0  412     0   412  763    70 84 16  0  0
2  0 207348  15756  78800 175424    0    0     0     0  874    25 89 11  0  0
```

```
1  0 207348  16368  78800 175596   0    0     0     0  940     24 86 14  0  0
1  0 207348  16600  78800 175604   0    0     0     0  929     27 95  3  0  2
3  0 207348  16976  78548 175876   0    0     0  2508  969     35 93  7  0  0
4  0 207348  16216  78548 175704   0    0     0     0  874     36 93  6  0  1
4  0 207348  16424  78548 175776   0    0     0     0  850     26 77 23  0  0
2  0 207348  17496  78556 175840   0    0     0     0  736     23 83 17  0  0
0  0 207348  17680  78556 175868   0    0     0     0  861     21 91  8  0  1
```

根据观察值，我们可以得出以下结论。

- 有大量的中断（in）和较少的上下文切换（cs）。这意味着一个单一的进程在产生对硬件设备的请求。
- 进一步显示某单个应用，user time（us）经常在 85%或者更多。考虑到较少的上下文切换，这个应用应该还在处理器中被处理。
- 运行队列还在可接受的性能范围内，其中有两个地方，是超出了允许限制。

**例子 2**：在下面的例子中，内核调度中的上下文切换处于饱和：

```
# vmstat 1
procs                    memory      swap          io     system         cpu
r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs us sy wa id
2  1 207740  98476  81344 180972   0    0  2496     0  900  2883  4 12 57 27
0  1 207740  96448  83304 180984   0    0  1968   328  810  2559  8  9 83  0
0  1 207740  94404  85348 180984   0    0  2044     0  829  2879  9  6 78  7
0  1 207740  92576  87176 180984   0    0  1828     0  689  2088  3  9 78 10
2  0 207740  91300  88452 180984   0    0  1276     0  565  2182  7  6 83  4
3  1 207740  90124  89628 180984   0    0  1176     0  551  2219  2  7 91  0
4  2 207740  89240  90512 180984   0    0   880   520  443   907 22 10 67  0
5  3 207740  88056  91680 180984   0    0  1168     0  628  1248 12 11 77  0
4  2 207740  86852  92880 180984   0    0  1200     0  654  1505  6  7 87  0
6  1 207740  85736  93996 180984   0    0  1116     0  526  1512  5 10 85  0
0  1 207740  84844  94888 180984   0    0   892     0  438  1556  6  4 90  0
```

根据观察值，我们可以得出以下结论。

- 上下文切换数目高于中断数目，说明 kernel 中相当数量的时间都开销在上下文切换线程。
- 大量的上下文切换将导致 CPU 利用率分类不均衡。很明显，实际上等待 IO 请求的百分比（wa）非常高，而 user time 百分比（us）非常低。
- 因为 CPU 都阻塞在 IO 请求上，所以运行队列里也有相当数目的可运行状态线程在等待执行。

### 15.2.5 使用 mpstat 工具进行多处理器监控

如果系统运行在多处理器芯片上，可以使用 mpstat 命令来监控每个独立的芯片。Linux 内核视双核处理器为两个 CPU，因此一个双核处理器的双内核就报告有 4 个 CPU 可用，这个需要引起注意。

mpstat 命令给出的 CPU 利用率统计值大致和 vmstat 一致，但是 mpstat 可以给出基于单个处理器的统计值：

```
# mpstat -P ALL 1
Linux 2.4.21-20.ELsmp (localhost.localdomain)  08/13/2012
05:17:31 PM  CPU   %user   %nice %system   %idle    intr/s
05:17:32 PM  all    0.00    0.00    3.19   96.53     13.27
05:17:32 PM    0    0.00    0.00    0.00  100.00      0.00
05:17:32 PM    1    1.12    0.00   12.73   86.15     13.27
05:17:32 PM    2    0.00    0.00    0.00  100.00      0.00
05:17:32 PM    3    0.00    0.00    0.00  100.00      0.00
```

下面给出一些具体的例子来介绍如何使用 mpstat 进行监控。

**例子 1**：在这个例子中，4 CPU 核心可用其中两个 CPU 主要处理进程运行（CPU 0 和 1）。第 3 个核心处理所有内核和其他系统功能（CPU 3）。第 4 个核心处于 idle（CPU 2）。

使用 top 命令可以看到有 3 个进程差不多完全占用了整个 CPU 核心：

```
# top -d 1
top - 23:08:53 up  8:34,   3 users,   load average: 0.91,  0.37,  0.13
Tasks: 190 total,    4 running,  186 sleeping,    0 stopped,    0 zombie
Cpu(s): 75.2% us, 0.2% sy, 0.0% ni, 24.5% id, 0.0% wa, 0.0% hi, 0.0%
Si
Mem:   2074736k total,    448684k used,   1626052k free,     73756k buffers
Swap:  4192956k total,          0k used,   4192956k free,    259044k cached
PID USER       PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
15957 nobody    25   0  2776  280  224 R  100  20.5  0:25.48 php
15959 mysql     25   0  2256  280  224 R  100  38.2  0:17.78 mysqld
15960 apache    25   0  2416  280  224 R  100  15.7  0:11.20 httpd
15901 root      16   0  2780 1092  800 R    1  0.1   0:01.59 top
1 root      16   0  1780  660  572 S    0  0.0   0:00.64 init
# mpstat -P ALL 1
Linux 2.4.21-20.ELsmp (localhost.localdomain)  08/13/2012
05:17:31 PM  CPU   %user   %nice %system   %idle    intr/s
05:17:32 PM  all   81.52    0.00   18.48   21.17    130.58
```

```
05:17:32 PM    0   83.67    0.00   17.35    0.00    115.31
05:17:32 PM    1   80.61    0.00   19.39    0.00     13.27
05:17:32 PM    2    0.00    0.00   16.33   84.66      2.01
05:17:32 PM    3   79.59    0.00   21.43    0.00      0.00
05:17:32 PM  CPU   %user   %nice %system   %idle    intr/s
05:17:33 PM  all   85.86    0.00   14.14   25.00    116.49
05:17:33 PM    0   88.66    0.00   12.37    0.00    116.49
05:17:33 PM    1   80.41    0.00   19.59    0.00      0.00
05:17:33 PM    2    0.00    0.00    0.00  100.00      0.00
05:17:33 PM    3   83.51    0.00   16.49    0.00      0.00
05:17:33 PM  CPU   %user   %nice %system   %idle    intr/s
05:17:34 PM  all   82.74    0.00   17.26   25.00    115.31
05:17:34 PM    0   85.71    0.00   13.27    0.00    115.31
05:17:34 PM    1   78.57    0.00   21.43    0.00      0.00
05:17:34 PM    2    0.00    0.00    0.00  100.00      0.00
05:17:34 PM    3   92.86    0.00    9.18    0.00      0.00
05:17:34 PM  CPU   %user   %nice %system   %idle    intr/s
05:17:35 PM  all   87.50    0.00   12.50   25.00    115.31
05:17:35 PM    0   91.84    0.00    8.16    0.00    114.29
05:17:35 PM    1   90.82    0.00   10.20    0.00      1.02
05:17:35 PM    2    0.00    0.00    0.00  100.00      0.00
05:17:35 PM    3   81.63    0.00   15.31    0.00      0.00
```

还可以使用 ps 命令通过查看 PSR 列，来检查哪个进程占用了哪个 CPU：

```
# while :; do  ps -eo pid, ni, pri, pcpu, psr, comm | grep ‘mysqld’; sleep 1;
Done
PID  NI PRI %CPU PSR COMMAND
15775   0  15 86.0   3 mysqld
PID  NI PRI %CPU PSR COMMAND
15775   0  14 94.0   3 mysqld
PID  NI PRI %CPU PSR COMMAND
15775   0  14 96.6   3 mysqld
PID  NI PRI %CPU PSR COMMAND
15775   0  14 98.0   3 mysqld
PID  NI PRI %CPU PSR COMMAND
15775   0  14 98.8   3 mysqld
PID  NI PRI %CPU PSR COMMAND
15775   0  14 99.3   3 mysqld
```

### 15.2.6 CPU监控总结

监控CPU性能由以下几个部分组成。

（1）检查system的运行队列，以及确定不要超出每个处理器3个可运行状态线程的限制。

（2）确定CPU利用率中user/system比例维持在70/30。

（3）当CPU开销更多的时间在system mode，那就说明已经超负荷并且应该尝试重新调度优先级。

（4）当I/O处理得到增长，CPU范畴的应用处理将受到影响。

## 15.3 内存监控详解

### 15.3.1 Virtual Memory介绍

虚拟内存就是采用硬盘对物理内存进行扩展，所以对可用内存的增加是要相对在一个有效范围内的。内核会写当前未使用内存块的内容到硬盘上，此时这部分内存被用于其他用途。当再一次需要原始内容时，此时再读回到内存中，这对于用户来说是完全透明的。在Linux下运行的程序能够看到，也仅仅是大量的可用内存，同时也不会留意到，偶尔还有部分是驻留在磁盘上的。当然，在硬盘上进行读和写，都是很慢的（大约会慢上千倍），相对于使用真实内存来说。因此程序无法运行得更快。用硬盘的一部分作为Virtual Memory，被称为“swap space”（交换空间）。

### 15.3.2 Virtual Memory Pages

虚拟内存被分为很多pages（页），在x86架构中，每个虚拟内存页为4KB。当内核写内存到磁盘或者读磁盘到内存，这就是一次写内存到页的过程。内核通常是在swap分区和文件系统之间进行这样的操作。

### 15.3.3 Kernel Memory Paging

内存分页在正常情况下总是活跃的，与Memory Swapping（内存交换）之间不要搞错了。内存分页是指内核会定期将内存中的数据同步到硬盘，这个过程就是Memory Paging。日复一日，应用最终将会消耗掉所有的内存空间。考虑到这点，内核必须经常扫描内存空间并且收回其中未被使用的内存页，然后再重新分配内存空间给其他应用使用。

### 15.3.4 kswapd

kswapd进程负责确保内存空间总是在被释放中。它监控内核中的pages_high和pages_low阈值，如果空闲内存的数值低于pages_low，则每次kswapd进程启动扫描并尝试释放32个free pages并一直重复该过程，直到空闲内存的数值高于pages_high。

kswapd进程完成以下几个操作。

- 如果该页处于未修改状态，则将该页放置回空闲列表中。

- 如果该页处于已修改状态并可备份回文件系统，则将页内容写入到磁盘。
- 如果该页处于已修改状态但没有任何磁盘备份，则将页内容写入到 swap device。

### 15.3.5 使用 vmstat 进行内存监控

vmstat 工具报告里除了 CPU 的使用情况，还包括了虚拟内存。以下就是 vmstat 输出中关于虚拟内存的部分。

- swapd：当前虚拟内存使用的总额（单位：KB）。空闲内存达到最低的阈值时，更多的数据被转换成页到交换设备中。
- free：当前内存中可用空间字节数。
- buff：当前内存中用于 read()和 write()操作的缓冲区中缓存字节数。
- cache：当前内存中映射到进程地址空间字节数。
- so：写入交换空间的字节数总额。
- si：从交换空间写回内存的字节数总额。
- bo：磁盘块页面从内存到文件或交换设备的总额。
- bi：磁盘块页面从文件或交换设备到内存的总额。

以下 vmstat 的输出结果，就是演示一个在 I/O 应用中，虚拟内存在高负荷情况下的环境：

```
# vmstat 3
procs memory swap io  system cpu
r b swpd free buff cache si so bi  bo in cs  us  sy id wa
3 2 809192 261556 79760 886880 416 0 8244 751 426 863 17 3 6 75
0 3 809188 194916 79820 952900 307 0 21745 1005 1189 2590 34 6 12 48
0 3 809188 162212 79840 988920 95 0 12107 0 1801 2633 2 2 3 94
1 3 809268 88756 79924 1061424 260 28 18377 113 1142 1694 3 5 3 88
1 2 826284 17608 71240 1144180 100 6140 25839 16380 1528 1179 19 9 12 61
2 1 854780 17688 34140 1208980 1 9535 25557 30967 1764 2238 43 13 16 28
0 8 867528 17588 32332 1226392 31 4384 16524 27808 1490 1634 41 10 7 43
4 2 877372 17596 32372 1227532 213 3281 10912 3337 678 932 33 7 3 57
1 2 885980 17800 32408 1239160 204 2892 12347 12681 1033 982 40 12 2 46
5 2 900472 17980 32440 1253884 24 4851 17521 4856 934 1730 48 12 13 26
1 1 904404 17620 32492 1258928 15 1316 7647 15804 919 978 49 9 17 25
4 1 911192 17944 32540 1266724 37 2263 12907 3547 834 1421 47 14 20 20
1 1 919292 17876 31824 1275832 1 2745 16327 2747 617 1421 52 11 23 14
5 0 925216 17812 25008 1289320 12 1975 12760 3181 772 1254 50 10 21 19
0 5 932860 17736 21760 1300280 8 2556 15469 3873 825 1258 49 13 24 15
```

根据观察值，我们可以得出以下结论。

- 大量的 disk pages（bi）被写入内存，很明显在进程地址空间里，数据缓存（cache）也在不断的增长。

- 在这个时间点上，空闲内存（free）始终保持在17MB，即使数据从硬盘读入而在消耗RAM。
- 为了维护空闲列表，kswapd 从读/写缓存区（buff）中获取内存并分配到空闲列表里，很明显可以看到buffer cache（buff）在逐渐的减少中。
- 同时kswapd进程不断的写脏页到swap device（so）时，很明显虚拟内存的利用率是在逐渐的增加中（swpd）。

### 15.3.6 内存监控总结

监控虚拟内存性能由以下几个部分组成。

- 当系统中出现较少的页错误，获得最好的响应时间，是因为memory caches（内存高速缓存）比disk caches（磁盘高速缓存）更快。
- 较少的空闲内存，是件好事情，那意味着缓存的使用更有效率。除非在不断地写入swap device和disk。
- 如果系统不断报告：swap device总是繁忙中，那就意味着内存已经不足，需要升级。

## 15.4 I/O 监控详解

### 15.4.1 I/O 监控介绍

磁盘I/O子系统是Linux系统中最慢的部分，主要是因为CPU到物理操作磁盘之间的距离（盘片旋转以及寻道）。

### 15.4.2 读和写数据——内存页

Linux内核将硬盘I/O进行分页，多数Linux系统的默认页大小为4KB。读和写磁盘块进出到内存都为4KB页大小。可以使用time命令加-v参数来检查系统中设置的页大小：

```
# /usr/bin/time -v date
<snip>
Page size (bytes): 4096
<snip>
```

### 15.4.3 Major and Minor Page Faults（主要页错误和次要页错误）

Linux类似多数的UNIX 系统，使用一个虚拟内存层来映射硬件地址空间。当一个进程被启动，内核先扫描CPU caches和物理内存。如果进程需要的数据在这两个地方都没找到，就需要从磁盘上读取，此时内核过程就是major page fault（MPF）。MPF要求磁盘子系统检索页并缓存进RAM。

一旦内存页被映射进内存的buffer cache（buff）中，内核将尝试从内存中读取或写入，此时内核过程就是minor page fault（MnPF）。与在磁盘上操作相比，MnPF通过反复使用内存中的内存页大大缩短了内核时间。

下面使用 time 命令验证当进程启动后 MPF 和 MnPF 的变化情况。在第一次运行进程时，MPF 会更多：

```
# /usr/bin/time -v evolution
<snip>
Major (requiring I/O) page faults: 163
Minor (reclaiming a frame) page faults: 5918
<snip>
```

第二次再运行时，内核已经不需要进行 MPF 了，因为进程所需的数据已经在内存中：

```
# /usr/bin/time -v evolution
<snip>
Major (requiring I/O) page faults: 0
Minor (reclaiming a frame) page faults: 5581
<snip>
```

### 15.4.4 The File Buffer Cache（文件缓存区）

文件缓存区就是内核将 MPF 过程最小化，而将 MnPF 过程最大化。随着系统不断地产生 I/O，buffer cache 也将不断地增加，直到内存不够，以及系统需要释放老的内存页去给其他用户进程使用时，系统就会丢弃这些内存页。结果是很多系统管理员对系统中过少的 free memory（空闲内存）表示担心，实际上这是系统更高效地在使用 caches。

下面来查看/proc/meminfo 文件：

```
# cat /proc/meminfo
MemTotal: 2075672 kB
MemFree: 52528 kB
Buffers: 24596 kB
Cached: 1766844 kB
<snip>
```

可以看出，该系统总计有 2GB（MemTotal）的可用内存，当前的空闲内存为 52MB（MemFree），有 24 MB 内存被分配磁盘写操作（Buffers），还有 1.7 GB 页用于读磁盘（Cached）。

### 15.4.5 Type of Memory Pages

在 Linux 内核中，Memory Pages 有三种，分别如下。

- Read Pages。这些页通过 MPF 从磁盘中读入，而且是只读。这些页存在于 Buffer Cache 中以及包括不能够修改的静态文件、二进制文件、库文件。当内核需要它们时，将读取到内存中；如果内存不足，内核将释放它们到空闲列表中。程序再次请求时，则通过 MPF 再次读回内存。
- Dirty Pages。这些页是内核在内存中已经被修改过的数据页。当这些页需要同步回磁盘上，由 pdflush 负责写回磁盘。如果内存不足，kswapd（与 pdflush 一起）将这些页写

回到磁盘上并释放更多的内存。

- Anonymous Pages。这些页属于某个进程，但是没有任何磁盘文件和它们有关。它们不能够同步回磁盘。如果内存不足，kswapd 将它们写入 swap 分区上并释放更多的内存。

### 15.4.6 Writing Data Pages Back to Disk

应用程序有很多选择可以写脏页回磁盘上，可通过 I/O 调度器使用 fsync()或 sync()系统函数来实现立即写回。如果应用程序没有调用以上函数，pdflush 进程会定期与磁盘进行同步。

### 15.4.7 监控 I/O

当觉得系统中出现了 I/O 瓶颈时，可以使用标准的监控软件来查找原因。这些工具包括前面介绍的 top、vmstat、iostat、sar。它们的输出结果一小部分是很相似的，不过每个都提供了各自对于性能不同方面的解释。

### 15.4.8 Calculating IO's Per Second（IOPS 的计算）

每个 I/O 请求到磁盘都需要若干时间，主要是因为磁盘的盘边必须旋转和机头必须寻道。磁盘的旋转常常被称为“rotational delay”（RD），机头的移动称为“disk seek”（DS）。一个 I/O 请求所需的时间计算就是 DS 加上 RD。磁盘的 RD 基于设备自身的 RPM 单位值（RPM 是 Revolutions Per-minute 的缩写，是转/每分钟，代表了硬盘的转速）。一个 RD 就是一个盘片旋转的半圆。那么如何计算一个 10K RPM 设备的 RD 值呢？主要分为以下几步。

（1）10000 RPM / 60 seconds（10000/60 = 166 RPS）；

（2）转换为 166 分之 1 的值（1/166 = 0.006 seconds/Rotation）；

（3）单位转换为毫秒（6ms/Rotation）；

（4）旋转半圆的时间（6/2 = 3ms），也就是 RD；

（5）加上平均 3ms 的寻道时间（3ms + 3ms = 6ms）；

（6）加上 2ms 的延迟（6ms + 2ms = 8ms）；

（7）1000ms / 8ms（1000/8 = 125 IOPS）。

每次应用程序产生一个 I/O，在 10K RPM 磁盘上都要花费平均 8ms。在这个固定时间里，磁盘将尽可能且有效地读写磁盘。IOPS 可以计算出大致的 I/O 请求数，10K RPM 磁盘有能力提供 120-150 次 IOPS。评估 IOPS 的效能可用每秒读写 I/O 字节数除以每秒读写 IOPS 数得出。

### 15.4.9 Random vs Sequential I/O（随机/顺序 I/O）

每个 I/O 产生的 KB 字节数是与系统本身 workload 相关的，有两种不同的 workload 类型，它们是 sequential 和 random。

#### 1. sequential I/O（顺序 IO）

iostat 命令提供的信息包括 IOPS 和每个 I/O 数据处理的总额，可使用 iostat -x 查看。顺序的 workload 是同时读顺序请求大量的数据。包括的应用有商业数据库（database）在执行大量的查询和流媒体服务。在这个 workload 中，KB per I/O 的比率应该是很高的。sequential workload 可

以同时很快地移动大量数据。如果每个 I/O 都节省了时间，那就意味着能带来更多的数据处理。

```
    # iostat -x 1
    avg-cpu: %user   %nice   %sys    %idle
    0.00      0.00    57.1 4  42.86
    Device:  rrqm/s wrqm/s   r/s  w/s    rsec/s  wsec/s    rkB/s  wkB/s
avgrq-sz avgqu-sz await   svctm  %util
    /dev/sda   0.00   12891.43  0.00  105.71  0.00  1   06080.00   0.00    53040.00
1003.46  1099.43  3442.43 26.49  280.00
    /dev/sda1 0.00  0.00     0.00 0.00   0.00    0.00      0.00   0.00
0.00     0.00     0.00    0.00   0.00
    /dev/sda2  0.00   12857.14  0.00  5.71    0.00     105782.86  0.00    52891.43
18512.00 559.14   780.00  490.00 280.00
    /dev/sda3 0.00  34.29    0.00 100.00 0.00    297.14    0.00   148.57
2.97     540.29   594.57  24.00  240.00
    avg-cpu: %user %nice %sys %idle
    0.00 0.00 23.53 76.47
    Device:  rrqm/s wrqm/s   r/s  w/s    rsec/s  wsec/s    rkB/s  wkB/s
avgrq-sz avgqu-sz await   svctm  %util
    /dev/sda   0.00   17320.59  0.00  102.94  0.00     142305.88  0.00    71152.94
1382.40  6975.29  952.29  28.57  294.12
    /dev/sda1 0.00  0.00     0.00 0.00   0.00    0.00      0.00   0.00
0.00     0.00     0.00    0.00   0.00
    /dev/sda2  0.00   16844.12  0.00  102.94  0.00     138352.94  0.00    69176.47
1344.00  6809.71  952.29  28.57  294.12
    /dev/sda3 0.00  476.47   0.00 0.00   0.00    952.94    0.00   1976.47
0.00     165.59   0.00    0.00   276.47
```

评估 IOPS 的效能，可用每秒读写 I/O 字节数除以每秒读写 IOPS 数得出，比如 rkB/s 除以 r/s 和 wkB/s 除以 w/s：

```
    53040/105 = 505KB per I/O
    71152/102 = 697KB per I/O
```

在上面的例子可以看出，每次循环下，/dev/sda 的 per I/O 都在增加。

#### 2．random I/O（随机 IO）

random 的 worklaod 环境下，不依赖于数据大小的多少，而更多的是依赖磁盘的 IOPS 数。Web 和 Mail 服务就是典型的 random workload，I/O 请求内容都很小。random workload 同时每秒会有更多的请求数产生，所以磁盘的 IOPS 数是关键，如下所示：

```
    # iostat -x 1
```

```
    avg-cpu: %user %nice %sys %idle
    2.04 0.00 97.96 0.00
    Device:  rrqm/s  wrqm/s r/s w/s rsec/s wsec/s rkB/s wkB/s avgrq-sz avgqu-sz
await svctm %util
    /dev/sda 0.00  633.67 3.06 102.31 24.49 5281.63 12.24 2640.82 288.89 73.67
113.89 27.22 50.00
    /dev/sda1 0.00   5.10  0.00 2.04  0.00  57.14   0.00   28.57   28.00  1.12
55.00 55.00 11.22
    /dev/sda2 0.00 628.57 3.06 100.27 24.49 5224.49 12.24 2612.24 321.50 72.55
121.25 30.63 50.00
    /dev/sda3 0.00   0.00  0.00  0.00 0.00   0.00  0.00  0.00       0.00  0.00
0.00  0.00  0.00
    avg-cpu: %user %nice %sys %idle
    2.15 0.00 97.85 0.00
    Device: rrqm/s wrqm/s r/s w/s  rsec/s wsec/s rkB/s wkB/s avgrq-sz avgqu-sz await
svctm %util
    /dev/sda   0.00    41.94    6.45   130.98   51.61   352.69   25.81   3176.34
19.79  2.90    286.32 7.37 15.05
    /dev/sda1 0.00 0.00   0.00   0.00  0.00   0.00  0.00    0.00  0.00  0.00
 0.00 0.00  0.00
    /dev/sda2   0.00   41.94    4.30   130.98   34.41   352.69   17.20   3176.34
21.18  2.90    320.00 8.24 15.05
    /dev/sda3 0.00 0.00   2.15   0.00 17.20   0.00  8.60    0.00  8.00  0.00
0.00 0.00  0.00
```

计算方式和之前的公式一致：

```
    2640/102 = 23KB per I/O
    3176/130 = 24KB per I/O
```

（对于顺序I/O来说，主要是考虑读取大量数据的能力，即KB per request。对于随机I/O系统，更需要考虑的是IOPS值）。

### 15.4.10 判断虚拟内存对I/O的影响

如果系统没有足够的RAM响应所有的请求，就会使用到SWAP device。就像使用文件系统I/O，使用SWAP device的代价也很大。如果系统已经没有物理内存可用，那就会在SWAP disk上创建很多很多的内存分页，如果同一文件系统的数据都在尝试访问SWAP device，那系统将遇到I/O瓶颈，最终导致系统性能的全面崩溃。如果内存页不能够及时读或写磁盘，它们就一直保留在RAM中。如果保留时间太久，内核又必须释放内存空间。因此，I/O操作都被阻塞住了，什么都没做就被结束了，不可避免地就出现kernel panic和system crash。

下面的 vmstat 示范了一个内存不足情况下的系统：

```
procs ——-memory——- —swap---io-- -system---cpu--
r  b    swpd   free  buff  cache   si   so    bi    bo   in cs   us sy id wa
17  0     1250  3248 45820 1488472    30 132   992    0 2437 7657 23 50  0 23
11  0     1376  3256 45820 1488888    57 245   416    0 2391 7173 10 90  0 0
12  0     1582  1688 45828 1490228    63 131  1348   76 2432 7315 10 90  0 10
12  2     3981  1848 45468 1489824   185 56   2300   68 2478 9149 15 12  0 73
14  2     10385 2400 44484 1489732     0 87   1112   20 2515 11620 0 12  0 88
14  2     12671 2280 43644 1488816    76 51   1812  204 2546 11407 20 45 0 35
```

从这个结果可以看出：大量的读请求回内存（bi），导致了空闲内存在不断地减少（free），这就使得系统写入 SWAP device 的块数目（so）和 swap 空间（swpd）在不断地增加。同时看到 CPU WIO time（wa）百分比很大。这表明 I/O 请求已经导致 CPU 开始效率低下。

要看 swaping 对磁盘的影响，可使用 iostat 检查 swap 分区，如下所示：

```
# iostat -x 1
avg-cpu: %user %nice %sys %idle
0.00  0.00 100.00 0.00
Device:   rrqm/s wrqm/s  r/s     w/s     rsec/s   wsec/s   rkB/s    wkB/s
avgrq-sz avgqu-sz await   svctm %util
/dev/sda  0.00   1766.67 4866.67 1700.00 38933.33 31200.00 19466.67 15600.00
10.68    6526.67  100.56  5.08  3333.33
/dev/sda1 0.00   933.33  0.00    0.00    0.00     7733.33  0.00     3866.67
0.00     20.00    2145.07 7.37  200.00
/dev/sda2  0.00    0.00     4833.33  0.00     38666.67  533.33    19333.33
266.67   8.11     373.33   8.07    6.90  87.00
/dev/sda3 0.00    833.33  33.33    1700.00 266.67    22933.33 133.33   11466.67
13.38    6133.33  358.46  11.35 1966.67
```

在这个例子中，swap device（/dev/sda1）和 file system device（/dev/sda3）在互相作用于 I/O。其中任意一个会有很高的写请求（w/s），也会有很高的 wait time（await），或者较低的服务时间比率（svctm），这表明两个分区之间互有联系，互有影响。

## 15.4.11　I/O 监控总结

I/O 性能监控包含了以下几点。

- 当 CPU 有等待 I/O 的情况时，那说明磁盘处于超负荷状态。
- 计算你的磁盘能够承受多大的 IOPS 数。
- 确定你的应用是属于随机或者顺序读取磁盘。
- 监控磁盘慢需要比较 wait time（await）和 service time（svctm）。

监控 SWAP 和系统分区，要确保 Virtual Memory 不是文件系统 I/O 的瓶颈。

# 第 16 章

# 见微知著：企业级 Linux 网络监控

## 本章导读

500 强企业除了需要具备系统和设备的监控功能外，还需要对其网络使用状况进行周密地监控、处理和优化，才能切实保证企业信息安全。在企业级 Linux 操作系统中，提供了许多优秀的开源网络监控工具辅助网络管理人员和信息安全工作人员来进行网络监控和管理。因此，本章将挑选一个企业应用最为广泛和稳定的 Cacti 工具来进行介绍。

## 16.1 Cacti 网络监控工具简介

Cacti（cacti.net）是一个随着时间推移（时间序列数据）用图表显示系统和网络信息的网络监测工具，并提供一个全功能的 Web 界面，可以浏览和检查网络设备的实时性能。例如，可以配置 Cacti 来监控经过本地服务器上的网络端口、本地网络上的交换机和路由器端口的网络流量。Cacti 图形提供网络各个部分的流量级别信息。当网络速度很慢时，可以参考历史图表，查看是否发生了任何超出普通网络流量的事情，Cacti 可以收集 CPU 利用率、磁盘空间使用率、Web 服务器上的页面浏览量和本地网络上的几乎所有其他数据。

Cacti 收集随着时间推移的基线（典型值）数据，可以使用这些信息来增加对系统和网络实时行为的洞察力，并帮助解决问题。这些信息甚至可以预测未来可能发生的事情（例如，当磁盘很可能被占满时）。当安装并配置 Cacti 时，它会定期轮询网络设备以获取所需数据，并把数据存储在 RRD 文件，用于 RRDtool（轮循数据库工具，oss.oetiker.ch/rrdtool）。Cacti Web 界面允许浏览设备和图形列表，并可看到随着时间推移的设备的可视化表示。

Cacti 是下一代监测工具的一部分。它是在以往工具吸取教训的基础上构建的，如 MRTG 和 Cricket 工具。这些工具都具有以下功能。

- 定期轮询跟踪设备数据。收集这些数据最常用的工具是 SNMP（简单网络管理协议，www.net - snmp.org）。
- 把数据存储在一个 RRD 文件。
- 具有 Web 界面，可以检查从存储数据生成的图表。这些图表通常显示每天、每周、每月和每年的信息。

不同的是：Cacti 的配置是通过其 Web 界面来实现的，而 MRTG 和 Cricket 的配置是通过编辑文本文件来实现的。

RRD 文件和 RRDtool 是 Cacti 许多功能的关键。Cacti 网站将 Cacti 描述为“完整的基于 RRDtool 的图形解决方案。”RRD 文件存储有效的时间序列数据，通过使用聚合功能，更容易保存最近时间段的大量细节信息，但逐渐减少 Cacti 文件中时间较长的细节数据。RRDtool 可以很容易地从 RRD 文件产生既简单又复杂的图形。Cacti 提供了许多扩展和插件。当你熟悉 Cacti 的基本使用和操作时，请访问 cacti.net/additional_scripts.php 以了解新增列表部分。还可以访问在同一站点的文档和用户论坛，获得更多关于 Cacti 的信息，以及学习如何为不同设备和数据源添加功能和支持。

Cacti 是用 PHP 语言实现的一个软件，它的主要功能是用 SNMP 服务获取数据，然后用 RRDtool 储存和更新数据，当用户需要查看数据的时候用 RRDtool 生成图表呈现给用户。因此，SNMP 和 RRDtool 是 Cacti 的关键。SNMP 关系着数据的收集，RRDtool 关系着数据存储和图表的生成；MySQL 配合 PHP 程序存储一些变量数据并对变量数据进行调用，如：主机名、主机 IP、SNMP 团体名、端口号、模板信息等变量；SNMP 抓到数据不是存储在 MySQL 中，而是存储在 RRDtool 生成的 RRD 文件中（在 Cacti 根目录的 RRA 文件夹下）。RRDool 对数据的更新和存储就是对 RRD 文件的处理，RRD 文件是大小固定的档案文件（Round Robin Archive），它能够存储的数据必须在创建时就已经定义。其工作架构和流程如图 16-1 和图 16-2 所示。

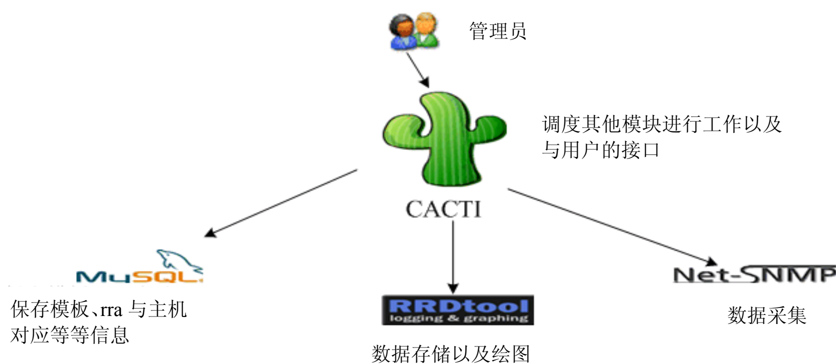



图 16-1 Cacti 的工作架构

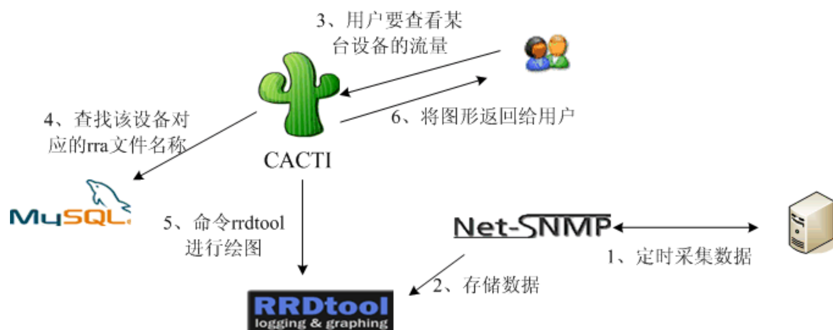



图 16-2 Cacti 的工作流程

## 16.2 安装和配置 Cacti

Cacti 的安装过程比较复杂，且安装的相关组件也比较多，包括 Apache、PHP、MySQL 等。下面对其主要的安装过程进行详细介绍。

### 16.2.1 安装辅助工具

1. 安装 MySQL

MySQL 的下载地址为：http://dev.mysql.com/downloads/mysql/5.0.html，按照以下步骤执行即可：

```
//查看系统中是否已经安装了 MySQL，如果是，卸载所有以 MySQL 开头的包。
# rpm -qa | grepmysql
# rpm -e mysql-*
//查找/etc/my.cnf（MySQL 的选项配置文件），如果有，请删除它，以免影响新安装版本的启动。
# rm -f /etc/my.cnf
# tar -zxvf mysql-standard-5.0.27-linux-i686-glibc23.tar.gz
# cp -rf mysql-standard-5.0.27-linux-i686-glibc23 /usr/local/
//建立符号链接，如果以后有新版本的 MySQL 的话，你可以仅仅将源码解压到新的路径，然后重新做
```

一个符号链接就可以了。这样非常方便，数据也更加安全。

```
# ln -s mysql-standard-5.0.27-linux-i686-glibc23 /usr/local/mysql
//添加用于启动 MySQL 的用户及用户组（如果以前安装过 MySQL，用户及用户组可能已存在）。
# useraddmysql
# groupaddmysql
//初始化授权表
# cd /usr/local/mysql
# scripts/mysql_install_db
//修改 MySQL 目录的所有权
# cd /usr/local
# chgrp -R mysql mysql-standard-5.0.27-linux-i686-glibc23
# chgrp -R mysql
# chown -R mysql mysql-standard-5.0.27-linux-i686-glibc23/data
# chown -R mysqlmysql/data
# ln -s /usr/local/mysql/bin/* /usr/local/bin/
//启动 MySQL
# bin/safe_mysqld  --user=mysql&
//配置系统启动时自动启动 MySQL
# cp support-files/mysql.server /etc/rc.d/init.d/mysqld
# chkconfig --add mysqld
//修改 MySQL 的最大连接数
# vi /etc/my.cnf
//添加以下行
[mysqld]
set-variable=max_connections=1000
set-variable=max_user_connections=500
set-variable=wait_timeout=200
//max_connections 设置最大连接数为 1000
//max_user_connections 设置每用户最大连接数为 500
//wait_timeout 表示 200 秒后将关闭空闲（IDLE）的连接，但是对正在工作的连接不影响。
//保存退出，并重新启动 MySQL
//重新启动 MySQL 后使用下面的命令查看修改是否成功
# mysqladmin -uroot -p variables
```

Password：

```
//可以看到以下项，说明修改成功
| max_connections              | 1000
```

```
| max_user_connections         | 500
| wait_timeout                 | 200
```

## 2. 安装Apache

Apache的下载地址为：http://httpd.apache.org/。

```
# tar -zxvfhttpd-2.2.4.tar.gz
# cd httpd-2.2.4
# ./configure --prefix=/usr/local/apache --enable-so
//编译时加上加载模块参数--enable-so
# make
# make install
#vi /usr/local/apache/conf/httpd.conf
//修改Apache配置文件，添加ServerNamewww.yourdomain.com（或ServerName本机IP）
# vi /etc/rc.d/rc.local
//在rc.local上加入一行/usr/local/apache/bin/apachectl -k start，系统启动时启动
Apache服务。
```

## 3. 安装PHP

先安装zlib、freetype、libpng、jpeg以便于让PHP支持GD库（Cacti的WeatherMap插件需要GD库的支持）。库文件下载地址：http://oss.oetiker.ch/rrdtool/pub/libs/。

（1）安装zlib。

```
# tar zlib-1.2.3.tar.gz
#cd zlib-1.2.3
# ./configure --prefix=/usr/local/zlib
#make
#make install
```

（2）安装libpng。

```
# tar zxvf libpng-1.2.16.tar.tar
#cd libpng-1.2.16
#cd scripts/
#mv makefile.linux ../makefile
#cd ..
#make
#make install
```

| 注意 | 这里的makefile不是用./configure生成，而是直接从scripts/里复制一个。 |
|---|---|

（3）安装freetype。

```
#tar zxvf freetype-2.3.4 .tar.gz
#cd freetype-2.3.4
#./configure --prefix=/usr/local/freetype
#make
#make install
```

（4）安装 jpeg。

```
# tar -zxf jpegsrc-1.v6b.tar.gz
# cd jpeg-6b/
# mkdir /usr/local/libjpeg
# mkdir /usr/local/libjpeg/include
# mkdir /usr/local/libjpeg/bin
# mkdir /usr/local/libjpeg/lib
# mkdir /usr/local/libjpeg/man
# mkdir /usr/local/libjpeg/man/man1
 //可以用mkdir -p /usr/local/libjpeg/man/man1 一步创建多层目录
# ./configure --prefix=/usr/local/libjpeg --enable-shared --enable-static
# make && make install
```

**注意** 这里 configure 一定要带--enable-shared 参数，不然，不会生成共享库。

（5）安装 fontconfig。

```
#tar -zxvf fontconfig-2.4.2.tar.gz
#cd fontconfig-2.4.2
#make#make install
```

（6）安装 GD。

```
#tar -zxvf gd-2.0.34.tar.gz
#cd gd-2.0.34
# ./configure --with-png --with-freetype=/usr/local/freetype --with-jpeg=/
usr/local/libjpeg
#make
#make install
```

编译时显示以下信息：

```
** Configuration summary for gd 2.0.34:
   Support for PNG library:          yes
   Support for JPEG library:         yes
   Support for Freetype 2.x library: yes
   Support for Fontconfig library:   yes
```

```
    Support for Xpm library:          no
    Support for pthreads:             yes
```

（7）编辑/etc/ld.so.conf。

添加以下几行到此文件中：

```
/usr/local/zlib/lib
/usr/local/freetype/lib
/usr/local/libjpeg/lib
/usr/local/libgd/lib
```

并执行 ldconfig 命令，使用动态装入器装载找到共享库。

（8）安装 libxml。

```
# tar -zxvf libxml2-2.6.25.tar.gz
    # cd libxml2-2.6.25
    # ./configure
    # make
    # make install
```

（9）安装 PHP。

PHP 下载地址：http://www.php.net/downloads.php#v5。

```
    #tar -zxvf  php-5.2.3.tar.gz
    #cd php-5.2.3
    #./configure --prefix=/usr/local/php --with-apxs2=/usr/local/apache2/bin/
apxs --with-mysql=/usr/local/mysql --with-gd=/usr/local/libgd --enable-gd-native
-ttf --with-ttf --enable-gd-jis-conv --with-freetype-dir=/usr/local/freetype
--with-jpeg-dir=/usr/local/libjpeg --with-png-dir=/usr --with-zlib-dir=/usr/
local/zlib --enable-xml --enable-mbstring --enable-sockets
    # make
    # make install
    # ln -s /usr/local/php/bin/* /usr/local/bin/
    # vi /usr/local/apache/conf/httpd.conf
```

查找 AddType application/x-compress .Z：

```
AddType application/x-gzip .gz .tgz
```

在其下加入 AddType application/x-tar .tgz：

```
AddType application/x-httpd-php .php
AddType image/x-icon .ico
```

修改 DirectoryIndex 行，添加 index.php。

修改为：

```
DirectoryIndexindex.php index.html index.html.var
```

```
# vi /usr/local/apache/htdocs/test.php
```

添加以下行：

```
<?php
    Phpinfo();
?>
```

保存退出。

```
# /usr/local/apache/bin/apachectl -k stop
#/usr/local/apache/bin/apachectl -k start
```

在浏览器中输入 http://www.yourdomain.com/test.php 进行测试。

对 PHP 编译选项的解释：

```
--prefix=/usr/local/php //指定 PHP 的安装目录
--with-apxs2=/usr/local/apache2/bin/apxs//支持 Apache 模块
--with-mysql=/usr/local/mysql            //支持 MySQL
--with-gd=/usr/local/libgd               //支持 GD 库
--enable-gd-native-ttf                   //激活对本地 TrueType 字符串函数的支持
--with-ttf                               //激活对 FreeType 1.x 的支持
--with-freetype-dir=/usr/local/freetype //激活对 FreeType 2.x 的支持
--with-jpeg-dir=/usr/local/libjpeg       //激活对 jpeg-6b 的支持
--with-png-dir                           //激活对 png 的支持
--with-zlib-dir=/usr/local/zlib          //激活对 zlib 的支持
--enable-mbstring                        //激活 mbstring 模块
--enable-gd-jis-conv                     //使 JIS-mapped 可用，支持日文字体
--with-mail                              //支持 Mail 函数
--enable-xml                             //支持 XML
--enable-sockets                         //支持套接字
```

（10）安装 RRDTool。

由于 rrdtool-1.2.23 需要一些库文件支持，故需先安装配置支持的环境，然后进行编译安装。直接运行以下 bash 脚本就可以完成安装：

> **注意**
>
> 将 cgilib-0.5.tar.gz、zlib-1.2.3.tar.gz、libpng-1.2.18.tar.gz、freetype-2.3.5.tar.gz、libart_lgpl-2.3.17.tar.gz、rrdtool-1.2.23.tar.gz 放到/root/rrdtool-1.2.23 目录下，将脚本保存为 /root/rrdtool-1.2.23/rrdtoolinstall.sh，并给执行权限 chmodu+x /root/rrdtool-1.2.23/rrdtoolinstall.sh。
>
> 以下链接是重新打好的一个 rrdtool-1.2.23 的安装包，里面包括了所有用到的库文件和安装脚本，下载解压后执行脚本 rrdinstall.sh 即可完成 RRDtool 的安装。点击下载 rrdtool-1.2.23.tar.gz。

```
#!/bin/sh
BUILD_DIR='pwd'
INSTALL_DIR=/usr/local/rrdtool
cd $BUILD_DIR
tar zxf cgilib-0.5.tar.gz
cd cgilib-0.5
make CC=gcc CFLAGS="-O3 -fPIC -I."
mkdir -p $BUILD_DIR/lb/include
cp *.h $BUILD_DIR/lb/include
mkdir -p $BUILD_DIR/lb/lib
cplibcgi* $BUILD_DIR/lb/lib

cd $BUILD_DIR
tar  zxf zlib-1.2.3.tar.gz
cd zlib-1.2.3
env CFLAGS="-O3 -fPIC" ./configure --prefix=$BUILD_DIR/lb
make
make install

cd $BUILD_DIR
tar zxvf libpng-1.2.18.tar.gz
cd libpng-1.2.18
env CPPFLAGS="-I$BUILD_DIR/lb/include" LDFLAGS="-L$BUILD_DIR/lb/lib" CFLAGS
="-O3 -fPIC" \
./configure --disable-shared --prefix=$BUILD_DIR/lb
Make
make install

cd $BUILD_DIR
tar zxvf freetype-2.3.5.tar.gz
cd freetype-2.2.5
env CPPFLAGS="-I$BUILD_DIR/lb/include" LDFLAGS="-L$BUILD_DIR/lb/lib" CFLAGS
="-O3 -fPIC" \
./configure --disable-shared --prefix=$BUILD_DIR/lb
Make
make install
```

```
    cd $BUILD_DIR
    tar zxvf libart_lgpl-2.3.17.tar.gz
    cd libart_lgpl-2.3.17
    env CFLAGS="-O3 -fPIC" ./configure --disable-shared --prefix=$BUILD_DIR/lb
    make
    make install

    IR=-I$BUILD_DIR/lb/include
    CPPFLAGS="$IR $IR/libart-2.0 $IR/freetype2 $IR/libpng"
    LDFLAGS="-L$BUILD_DIR/lb/lib"
    CFLAGS=-O3
    export CPPFLAGS LDFLAGS CFLAGS

    cd $BUILD_DIR
tar zxf rrdtool-1.2.23.tar.gz
    cd rrdtool-1.2.23
    ./configure --prefix=$INSTALL_DIR --disable-python --disable-tcl&& make &&
make install

    //完成后建立符号连接
    ln -s /usr/local/rrdtool/bin/* /usr/local/bin/
    //执行 RRDtool 看是否安装正确
```

（11）安装 net-snmp。

Red Hat 默认安装了 SNMP 服务，但好像没有 snmpwalk、snmpget 这两个命令，所以需要编译安装 NET-SNMP。NET-SNMP 官方网站为：http://www.net-snmp.org/。

```
    # tar zxvf net-snmp-5.2.4.tar.gz
    #cd net-snmp-5.2.4
    #./configure --prefix=/usr/local/net-snmp  --enable-developer
    #make
    #make install
    # ln -s /usr/local/net-snmp/bin/* /usr/local/bin/
    #cpEXAMPLE.conf  /usr/local/net-snmp/share/snmp/snmpd.conf

    //修改 snmpd.conf（修改 COMMUNITY、允许抓取 SNMP 数据的主机、抓取数据范围等）。
    # /usr/local/net-snmp/sbin/snmpd    //启动 SNMP 服务
```

```
# vi /etc/rc.d/rc.local

//在 rc.local 上加入一行/usr/local/net-snmp/sbin/snmpd,系统启动时启动 SNMP 服务。
```

### 16.2.2 安装 Cacti

#### 1. 基本安装

Cacti 的官方网站为：www.cacti.net/。执行以下命令：

```
# tar -zxvf cacti-0.8.6j.tar.gz
# mv -r cacti-0.8.6j /usr/loca/apache/htdocs/cacti
# vi /usr/local/apache/htdocs/cacti/include/config.php
$database_type = "mysql";
$database_default = "cacti";
$database_hostname = "localhost";
$database_username = "cacti";
$database_password = "cacti";
//添加 Cacti 用户
# useradd cacti
//将 rra 目录的所有权给 Cacti 用户
# chown -R cacti /usr/loca/apache/htdocs/cacti/rra
//修改 Cacti 目录所属组
# chgrp -R cacti /usr/loca/apache/htdocs/cacti
//为 Cacti 用户添加 cron 任务
# su - cacti
# crontab -e
*/5 * * * * /usr/local/bin/php /usr/local/apache/htdocs/cacti/poller.php>
/dev/null 2>&1
```

> **注意** 首次执行 poller.php 时请使用 Cacti 用户，否则生成的 rrd 文件 Cacti 将没有写入权限。

#### 2. 安装 Cactid

Cactid 的安装需要以下支持。

- net-snmp-devel（需要编译安装 net-snmp 时添加--enable-developer 选项）
- mysql
- mysql-devel（mysql 源文件编译安装后默认支持）
- openssl-devel（Red Hat 默认安装）

```
# tar -zxvf cacti-cactid-0.8.6i.tar.gz
# cd cacti-cactid-0.8.6i
# ./configure --with-mysql=/usr/local/mysql --with-snmp=/usr/local/net-snmp
# make
//这时将在此目录下看到多出了 cactid、cactid.conf 两个文件
# mkdir /usr/local/cactid
# cpcactidcactid.conf /usr/local/cactid
# vi  /usr/local/cactid/cactid.conf     //修改 cactid 配置文件
DB_Host         127.0.0.1
DB_Database     cacti
DB_User         cacti
DB_Pass         cacti
```

### 3. 数据库配置

数据库配置如下：

```
#mysql -uroot -p
Password:
mysql> create database cacti;
Query OK, 1 row affected (0.00 sec)

mysql> grant all on cacti.* to cacti@localhost identified by “cacti”;
Query OK, 1 row affected (0.00 sec)

mysql>exit
# cd /usr/local/apache/htdocs/cacti
# mysql -uroot -p cacti <cacti.sql
Password:
```

### 4. 完成 Cacti 的安装

主要包括如下几个步骤，即可完成 Cacti 的安装。

（1）在浏览器中输入 http://www.yourdomain.com/cacti/。

默认用户名：admin；密码：admin

（2）更改密码

（3）设置 Cacti 用到的命令路径，如图 16-3 所示。

```
snmpwalk Binary Path    /usr/local/ bin/snmpwalk
snmpget Binary Path     /usr/local/ bin/snmpget
RRDTool Binary Path     /usr/local/ bin/rrdtool
```

```
PHP Binary Path         /usr/local/bin/php
Cacti Log File Path     /usr/local/apache/htdocs/cacti/log/cacti.log
CactidPoller File Path   /usr/local/cactid/cactid
```

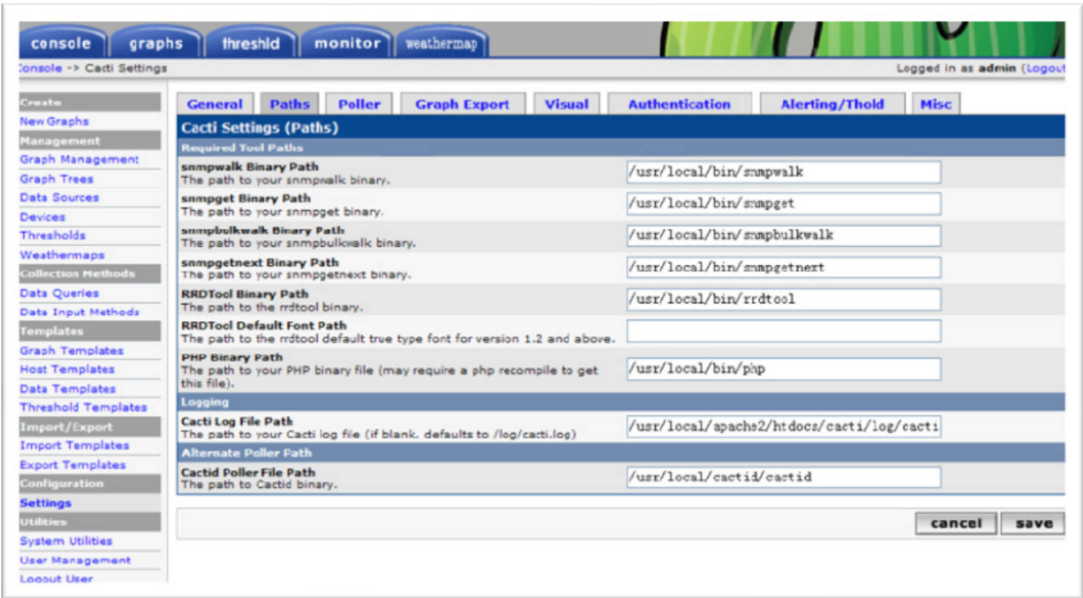



图 16-3　Cacti 的路径设置

（4）进入 Cacti 后需确认更改以下位置，如图 16-4 和图 16-5 所示。

```
Console>Settings>General
```

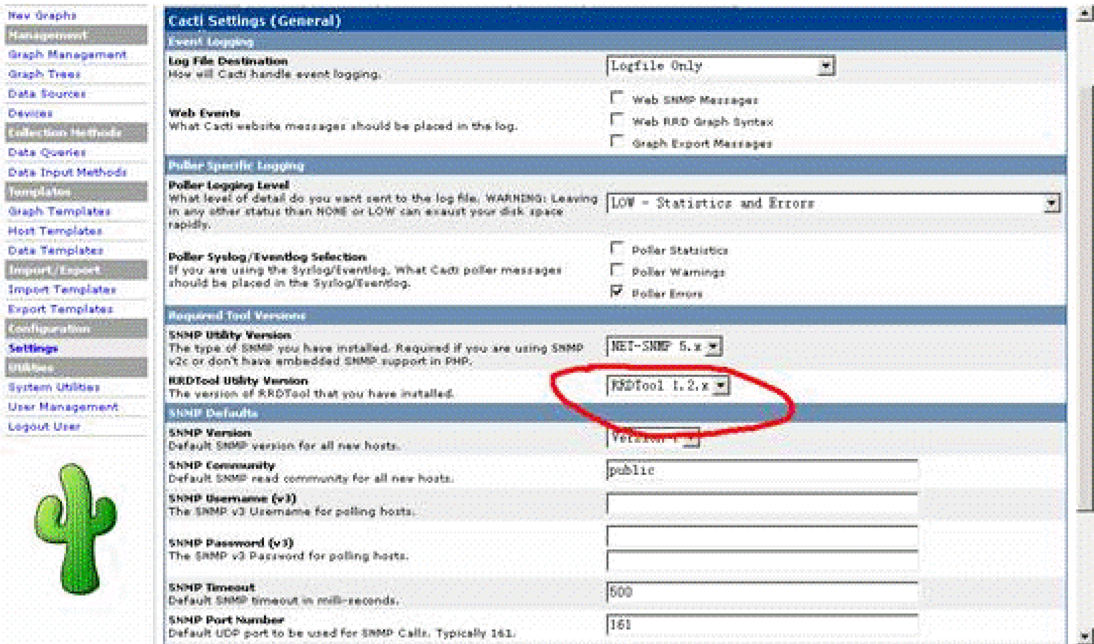



图 16-4　General 设置

```
Console>Settings>Poller
```

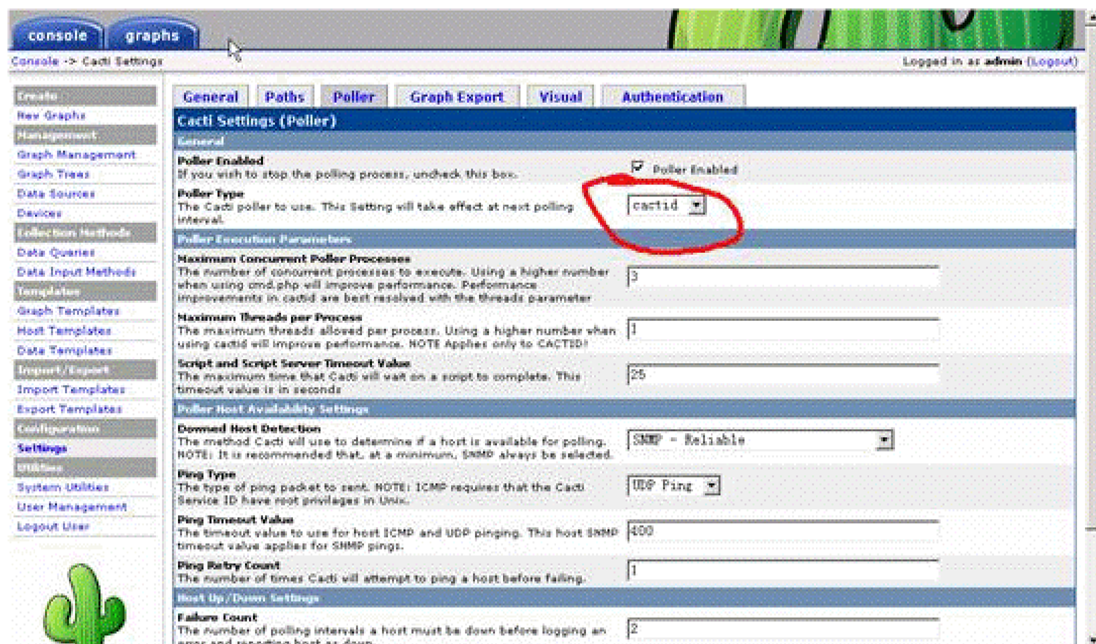



图 16-5　Poller 设置

# 16.3　使用 Cacti

## 16.3.1　Cacti 界面介绍

登录 Cacti 后，可以看到左上角是两个选项卡：console 和 graphs，如图 16-6 和图 16-7 所示。console 表示控制台，在此进行所有的配置等操作；而 graphs 则是用来查看所有服务器的性能图像的界面。

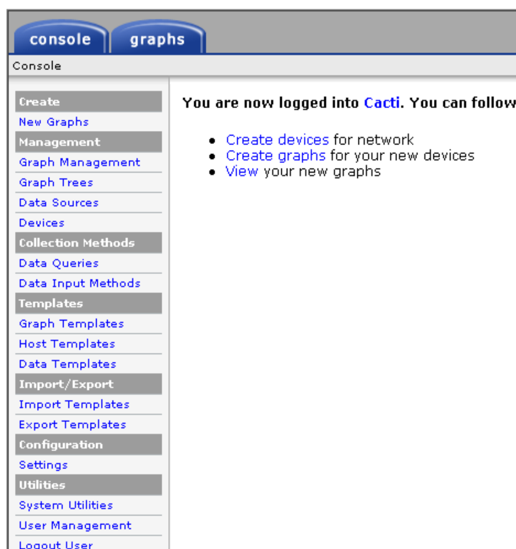



图 16-6　Cacti 主界面

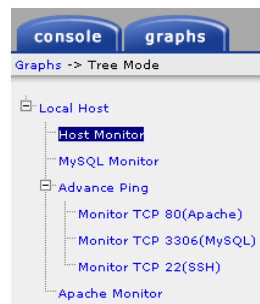



图 16-7　graphs 选项卡

console 选项卡有以下选项。

（1）Create。

New Graphs：创建新图像的快捷方式。

（2）Management。

Graph Management：图像管理。可以在此删除、复制图像，Cacti 会自动创建图像。不过如果我们有特殊的需要，比如将几张图上的数据合并在一张图像上的话也可以在此手工新建图像。

Graph Trees：图像树。在 graphs 界面里，图像或 devices 是树状结构显示的，可以在此设置树的结构。

Data Sources：管理 RRD 文件，如图 16-8 所示。一般无须修改，Cacti 会自己创建 RRD 文件。

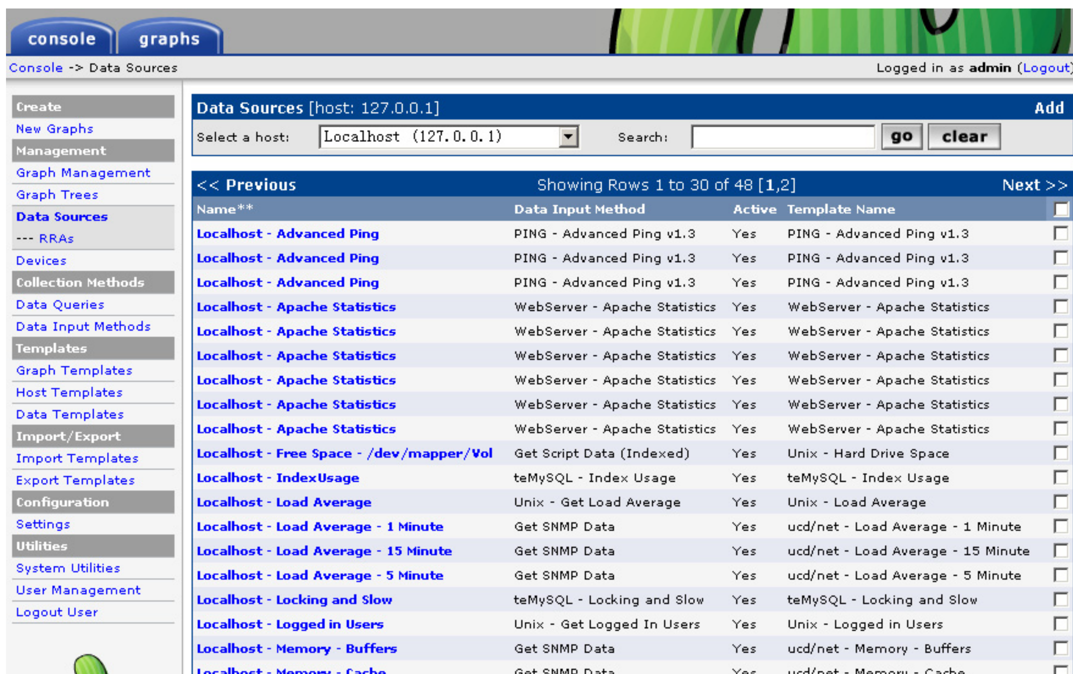



图 16-8　Data Sources 管理

Devices：设备管理。这是我们经常需要修改的地方，可以在此创建新的设备或修改其名称等信息。

（3）Collection Methods。

Data Queries 和 Data Input Methods 是采集数据的方式，一般我们无须对这两项进行修改。

（4）Templates。

Graph Templates、Host Templates 和 Data Templates 分别为图像模板、主机类型模板和数据模板。这些模板可以导出、导入也可以自己编写，一般无须修改。

（5）Import/Export。

Import Templates 和 Export Templates 是对上述模板的导入、导出。我们可以在 Cacti 的官方网站上找到这些模板，不过需要注意模板对应的 Cacti 版本。

（6）Configuration。

Settings：Cacti 的主要配置菜单。

可以在此重新设置对应的程序的路径、版本等信息，也可以设置图像的输出方式（允许 FTP）、显示效果、登录方式（允许使用 LDAP）等。

（7）Utilities。

System Utilities：显示 Cacti 系统的一些 cache 和 log 信息，如果 log 文件太大，建议直接到后台查看。

User Management：用户管理。可以在此添加、删除用户，并对每个用户设置详细的权限。

Logout User：注销用户。

## 16.3.2 创建监测点

假设被监测的服务器名叫"Test Host"，IP 为 192.168.100.110，SNMP 的 community 为 public。切换到 Cacti 的 console 选项卡，单击 Devices 选项，进入设备选项界面，单击 Add 链接添加新设备，如图 16-9 所示。

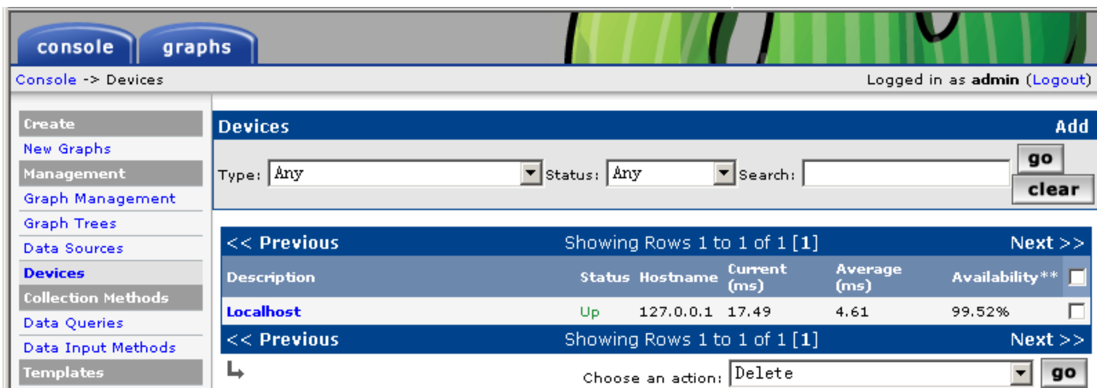



图 16-9　设置监测点

如图 16-10 所示，填写要监测服务器的各种信息，其中 Host Template 请选择 Local Linux Machine 或 ucd/net SNMP Host（选择一个合适的主机模板），单击 Create 按钮保存信息，如果 SNMP 连接没有问题，左上角会出现该服务器的信息，如图 16-11 所示，否则会出现"SNMP error"的红色字样。

图 16-10　添加新设备

图 16-11　SNMP 连接成功

单击图 16-11 右上角的“Create Graphs for this Host”，为该设备创建需监测的内容。监测的内容分两种：Graph Templates 和 Data Query，区别在于 Data Query 能根据 SNMP 信息列出监测

项目的信息。例如，Data Query 中的 Interface Statistics 可以看到该主机所有网卡的信息，这样我们可以选择需要监测的网卡。选中右侧的复选框，选择要监测的项目，如图 16-12 所示。

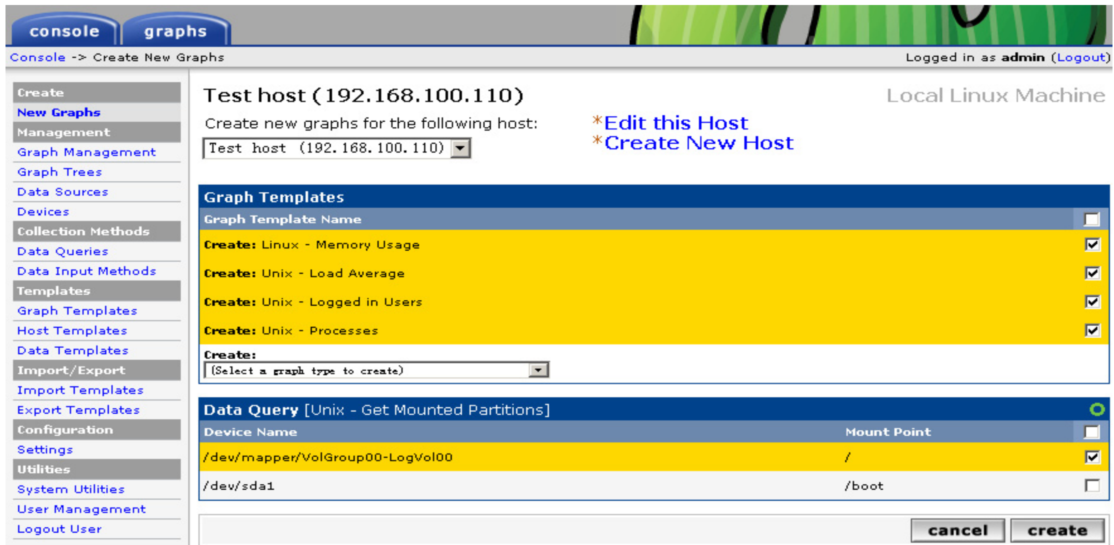



图 16-12　为设备创建需监测的内容

单击 Create 按钮创建选择的监测内容。已经选择创建的内容会自动变成灰色并且不能再进行选择。Cacti 会自动创建该监测点的 RRD 文件（在 rra 文件夹中）、Data Source 和 graph 条目，如图 16-13 所示。

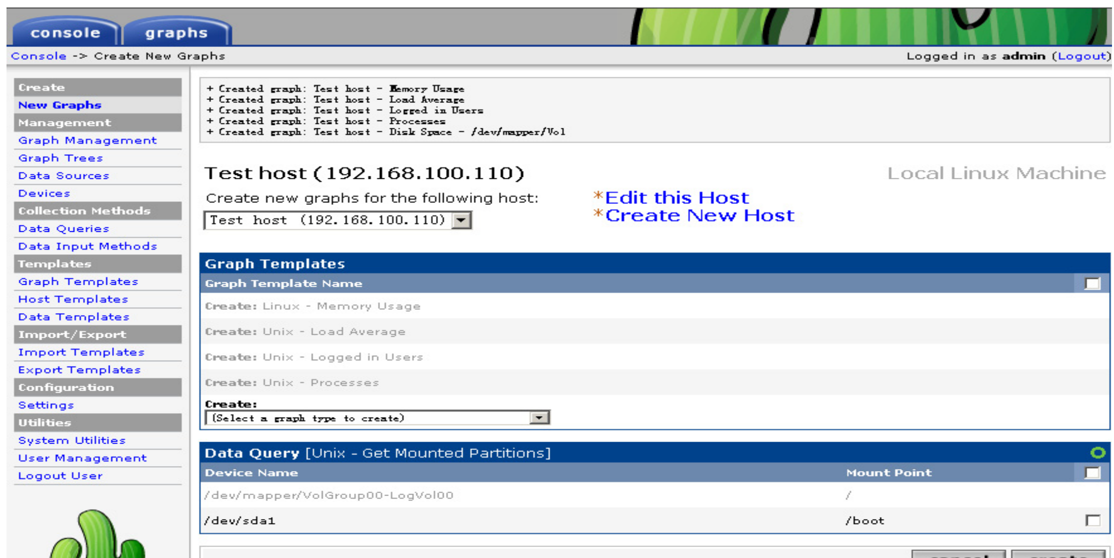



图 16-13　创建监测点完毕

## 16.3.3　查看监测点

单击 Graph Management 选项可以看到刚才创建的监测点对应的图像，如图 16-14 所示。注意由于 Cacti 默认每 5 分钟到监测服务器上取一次数据，所以刚创建的监测点会出现图像不能显

示的现象，需要等几分钟查看才会正常显示。




图 16-14　查看 Graph Management

为了方便查看，可以将刚才新创建的设备或图像加入到图像树上，单击 Graph Trees 选项，进入图像树选项界面，如图 16-15 所示。

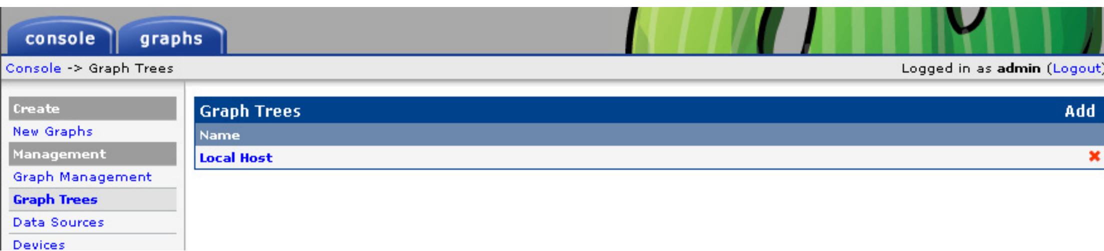



图 16-15　图像树界面

单击分支名称或 Add 链接添加新的分支，如图 16-16 所示。

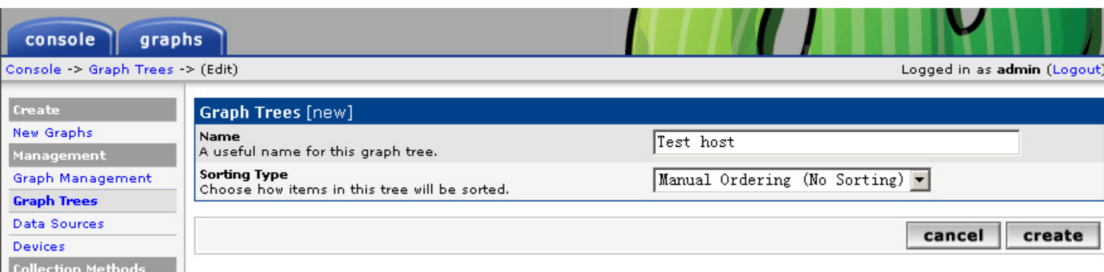



图 16-16　添加新分支

单击 Add 链接，添加新的 Tree Items，然后在 Tree Item Type 选项栏中选择 Host 选项，在 Tree Item Value 选项的 Host 下拉列表框中选择刚才新添加的主机“Test Host “，并单击 Create 按钮，如图 16-17 所示。

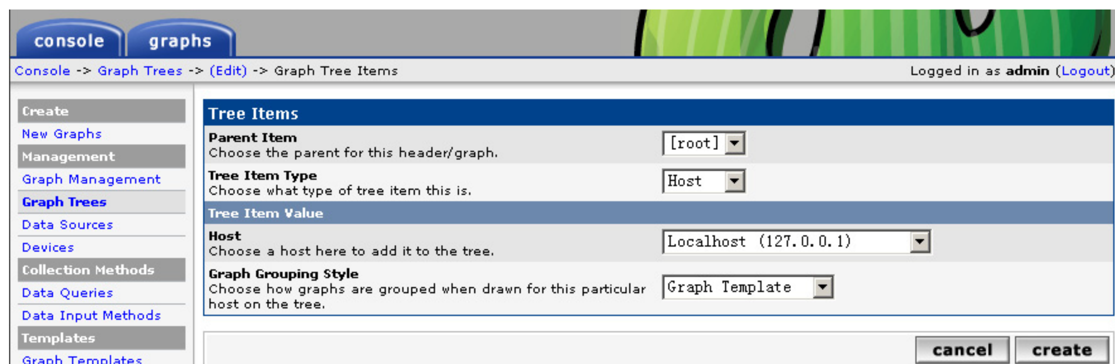



图 16-17　添加 Tree Items

直接在 ViewTree 中添加一个 Host 节点进行监控无疑是一个简单的方法，但随着监控图的增多，将所有的图像放到一个 Host 节点上，当查看图像时图像的显示速度会变慢，而且监控图的条理也不清楚，所以通常可以根据监控图监控的功能和监控服务的类型等进行分类，整理出一棵有条理的图像树。可以按下面的步骤添加节点。

（1）如图 16-18 所示，首先添加一个 root 节点，在 Tree Item Type 选项栏中选择 Header 选项，在 Title 文本框中填写一个合适的描述性信息，如本例填写“Host Monitor”，Host Monitor 节点的图像主要是和主机性能相关的一些图像，如 CPU、内存、磁盘空间等。

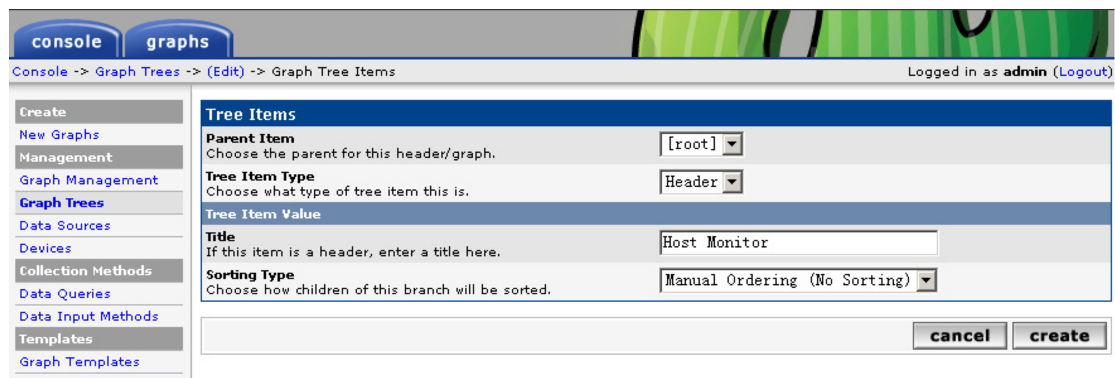



图 16-18　添加 root 节点

（2）单击刚创建的 root 节点（Host Monitor）后面的 Add 链接来添加一个 Graph，如图 16-19 所示。

Save Successful.

| Graph Trees [edit: Test host] | |
|---|---|
| **Name**<br>A useful name for this graph tree. | Test host |
| **Sorting Type**<br>Choose how items in this tree will be sorted. | Manual Ordering (No Sorting) |

Tree Items　Add

++ --

| Item | Value |
|---|---|
| Host: Localhost (127.0.0.1) | Host |
| ⊟ Host Monitor (Add) | Heading |

cancel　save

图 16-19　为 root 添加 Graph

（3）在 Tree Item Type 选项栏中选择 Graph 选项，在 Graph 下拉列表框中选择想要加入的监控图。重复此过程，加入所有你想加入的监控图，如图 16-20 所示。

| Tree Items | |
|---|---|
| **Parent Item**<br>Choose the parent for this header/graph. | --- Host Monitor |
| **Tree Item Type**<br>Choose what type of tree item this is. | Graph |
| **Tree Item Value** | |
| **Graph**<br>Choose a graph from this list to add it to the tree. | Localhost - CPU Usage |
| **Round Robin Archive**<br>Choose a round robin archive to control how this graph is displayed. | Daily (5 Minute Average) |

cancel　create

图 16-20　在 Graph 下拉列表框中选择想要加入的监控图

如图 16-21 所示，显示添加成功。

Save Successful.

| Graph Trees [edit: Test host] | |
|---|---|
| **Name**<br>A useful name for this graph tree. | Test host |
| **Sorting Type**<br>Choose how items in this tree will be sorted. | Manual Ordering (No Sorting) |

Tree Items　Add

++ --

| Item | Value |
|---|---|
| Host: Localhost (127.0.0.1) | Host |
| ⊟ Host Monitor (Add) | Heading |
| Localhost - CPU Usage | Graph |

cancel　save

图 16-21　添加成功

你还可以添加 Tree Items，如 MySQL Monitor，此节点用来监控与 MySQL 服务相关的信息，重复上面的过程，直到一棵有层次和条理的树创建完成，如图 16-22 所示。

Tree Items Add

| Item | Value |
|---|---|
| Host Monitor (Add) | Heading |
| MySQL Monitor (Add) | Heading |
| Localhost - teMySQL - Load Average | Graph |
| Localhost - teMySQL - CPU Usage | Graph |
| Localhost - teMySQL - Network Usage | Graph |
| Localhost - teMySQL - Index Usage | Graph |
| Localhost - teMySQL - Select Queries | Graph |
| Localhost - teMySQL - Select Handler | Graph |
| Localhost - teMySQL - Query Cache | Graph |
| Localhost - teMySQL - Locking and Slow | Graph |
| Localhost - teMySQL - Old Command Stats | Graph |
| Localhost - teMySQL - Sorts | Graph |
| Localhost - teMySQL - Threads/Abends | Graph |
| Localhost - teMySQL - Volatile Queries | Graph |
| Advance Ping (Add) | Heading |
| Monitor TCP 80(Apache) (Add) | Heading |
| Localhost - Advanced Ping | Graph |
| Monitor TCP 3306(MySQL) (Add) | Heading |
| Localhost - Advanced Ping | Graph |
| Monitor TCP 22(SSH) (Add) | Heading |
| Localhost - Advanced Ping | Graph |
| Apache Monitor (Add) | Heading |
| Localhost - Apache Statistics - Hits / s | Graph |
| Localhost - Apache Statistics - Bytes / Request | Graph |
| Localhost - Apache Statistics - kBits / s | Graph |
| Localhost - Apache Statistics - Thread Details | Graph |
| Localhost - Apache Statistics - Thread Details (%) | Graph |

图 16-22 创建成功的树

这样，我们就可以在 graphs 选项卡中查看 Test Host 的所有监测图像了，如图 16-23 所示。

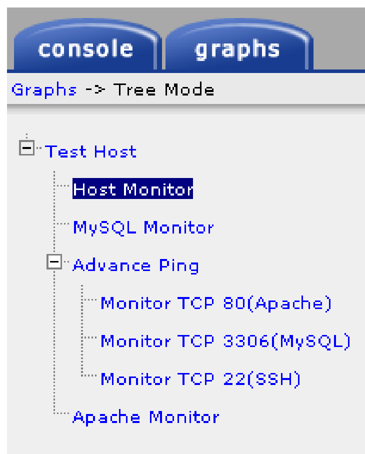



图 16-23 在 graphs 选项卡中查看所有监测图像

## 16.3.4 为已有 Host 添加新的监控图

在 console 控制台下单击 New Graphs 选项，选择要添加监控图的主机。在 Graph Templates 中选择一个 Graph 模板，本例选择 SNMP - Ceneric OID Template，单击 Create 按钮，如图 16-24 所示。

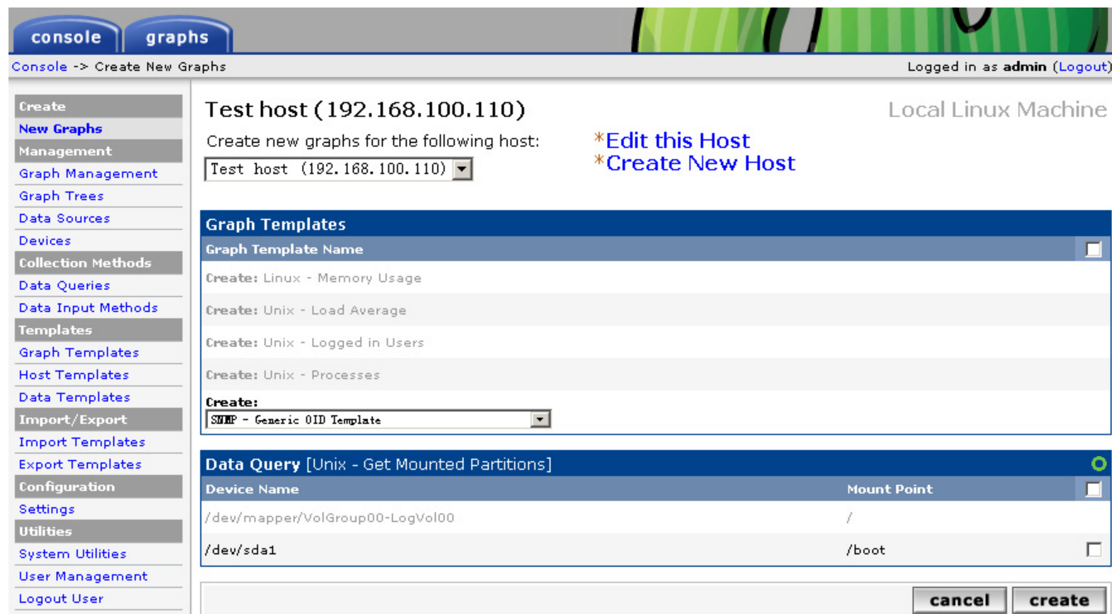



图 16-24 选择 Host

在 Title 文本框中填写 Graph 的名字，在 Vertical Lable 文本框中填写描述信息或是所绘图片使用的单位等，显示在所绘图片的左侧，在 Name 文本框中填写此数据源的名字，在 Legend Color 下拉列表框中选择画图使用的颜色，在 Legend Text 文本框中填写图例的名字，在 OID 文本框中填写要监控主机的 OID 信息，单击 Create 按钮完成 Graph 的创建，如图 16-25 所示。

| Create Graph from 'SNMP - Generic OID Template' | |
|---|---|
| **Graph** [Template: SNMP - Generic OID Template] | |
| **Title**<br>The name that is printed on the graph. | \|host_description\| - Tcp Established |
| **Vertical Label**<br>The label vertically printed to the left of the graph. | Established |
| **Graph Items** [Template: SNMP - Generic OID Template] | |
| **Legend Color**<br>The color to use for the legend. | 00FF00 |
| **Legend Text**<br>Text that will be displayed on the legend for this graph item. | Established |
| **Data Source** [Template: SNMP - Generic OID Template] | |
| **Name**<br>Choose a name for this data source. | \|host_description\| - Tcp Established |
| **Maximum Value [snmp_oid]**<br>The maximum value of data that is allowed to be collected. | 100 |
| **Data Source Type [snmp_oid]**<br>How data is represented in the RRA. | GAUGE |
| **Custom Data** [Template: SNMP - Generic OID Template] | |
| **OID** | .1.3.6.1.2.1.6.9.0 |

cancel create

图 16-25 填写相关信息

完成后监控图如图 16-26 所示。

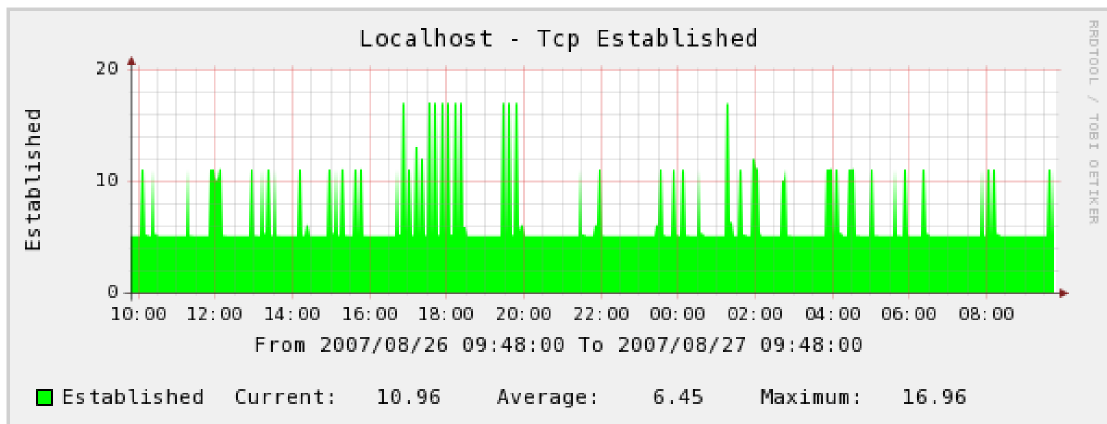



图 16-26　监控图示意

## 16.3.5　合并多个数据源到一张图上

在 console 控制台下单击 Graph Management 选项，然后单击 Add 链接。在 Selected Graph Template 下拉列表框中选择 None 选项，在 Host 下拉列表框中选择 None 选项，然后单击 Create 按钮。在 Title 文本框中输入 Graph 的名字，如图 16-27 所示。

**Graph Template Selection** [new]

| | |
|---|---|
| **Selected Graph Template**<br>Choose a graph template to apply to this graph. Please note that graph data may be lost if you change the graph template after one is already applied. | None |
| **Host**<br>Choose the host that this graph belongs to. | None |

**Graph Configuration**

| | |
|---|---|
| **Title**<br>The name that is printed on the graph. | TCP TEST |
| **Image Format**<br>The type of graph that is generated; GIF or PNG. | PNG |
| **Height**<br>The height (in pixels) that the graph is. | 120 |
| **Width**<br>The width (in pixels) that the graph is. | 500 |
| **Auto Scale**<br>Auto scale the y-axis instead of defining an upper and lower limit. Note: if this is check both the Upper and Lower limit will be ignored. | ☑ Auto Scale |
| **Auto Scale Options**<br>Use --alt-autoscale-max to scale to the maximum value, or --alt-autoscale to scale to the absolute minimum and maximum. | ○ Use --alt-autoscale<br>◉ Use --alt-autoscale-max |
| **Logarithmic Auto Scaling (--logarithmic)**<br>Use Logarithmic y-axis scaling | ☐ Logarithmic Auto Scaling (--logarithmic) |
| **Rigid Boundaries Mode (--rigid)**<br>Do not expand the lower and upper limit if the graph contains a value outside the valid range. | ☐ Rigid Boundaries Mode (--rigid) |
| **Auto Padding**<br>Pad text so that legend and graph data always line up. Note: this could cause graphs to take longer to render because of the larger overhead. Also Auto Padding may not be accurate on all types of graphs, consistant labeling usually helps. | ☑ Auto Padding |
| **Allow Graph Export**<br>Choose whether this graph will be included in the static html/png export if you use cacti's export feature. | ☑ Allow Graph Export |

图 16-27　填写相关信息

单击 Add 链接，添加 Graph Items，如图 16-28 所示。

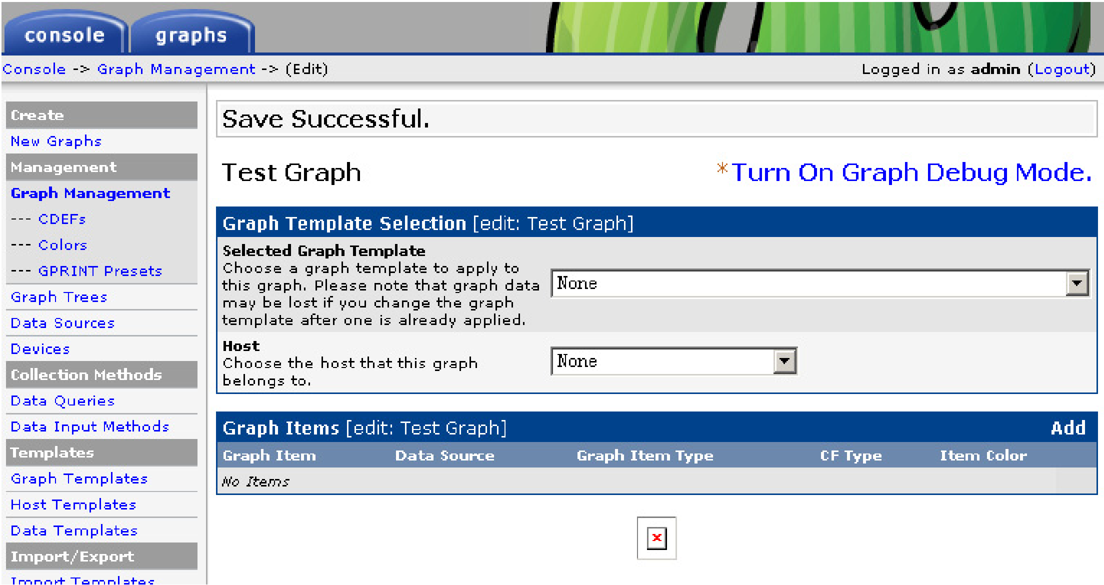



图 16-28　添加 Graph Items

添加多个数据源，将所选的多个数据源画到一张监控图上，如图 16-29 所示。

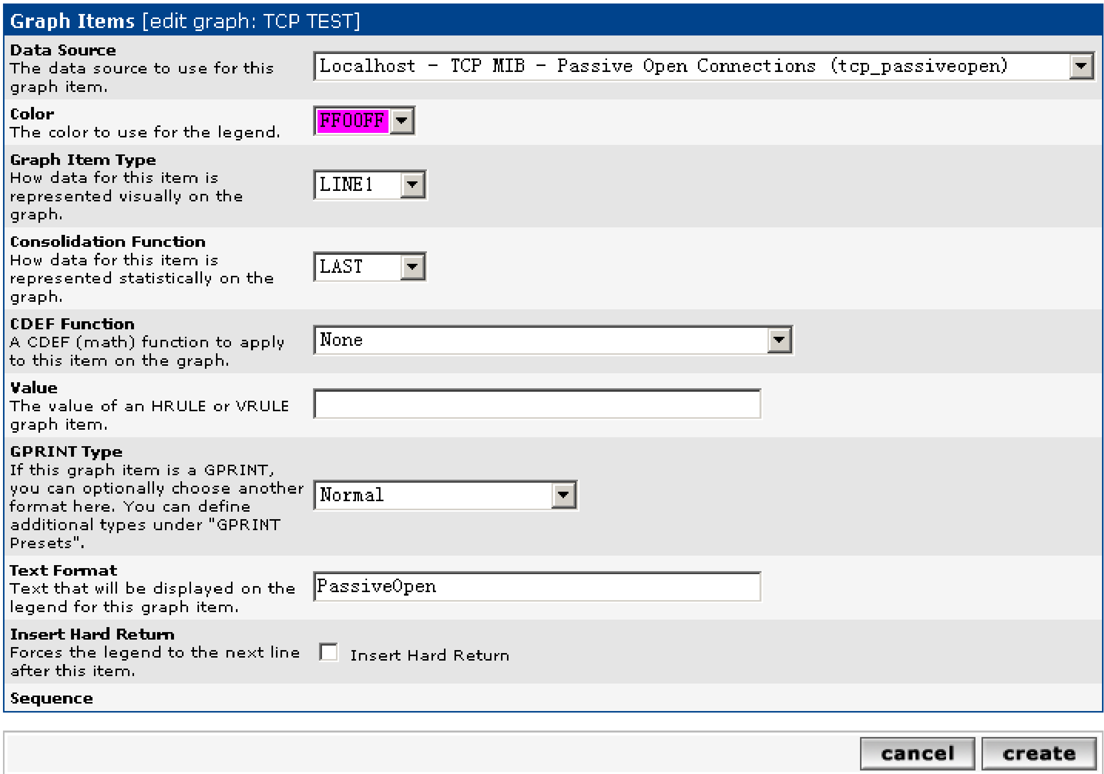



图 16-29　添加多个数据源

完成后如图 16-30 所示。

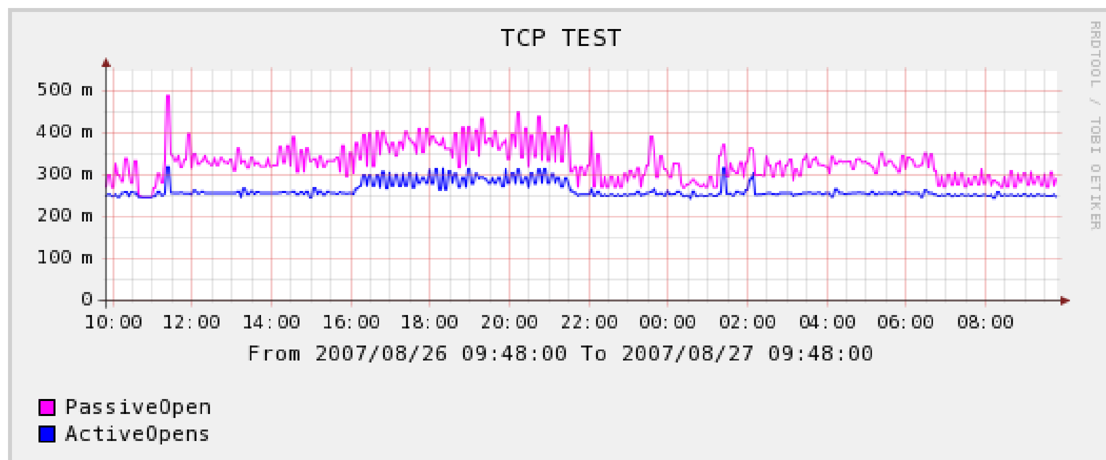



图 16-30　设置成功后的显示

## 16.3.6　使用 Cacti 插件

Cacti 插件是对 Cacti 的扩展。要使用 Cacti 插件必须先扩展 Cacti 架构，以此来支持插件。主要要完成以下一些工作。

### 1．安装 Cacti 插件架构扩展

下载地址：http://cactiusers.org/downloads/patches/。然后执行以下指令。

（1）解压下载的 tar 包，你会得到一个 cacti-plugin-arch 目录：

```
# tar -zxvf cacti-plugin-arch.tar.gz
# cd cacti-plugin-arch
# ls
[root@lib cacti-plugin-arch]# ls
Cacti-plugin-0.8.6i.diff  files-0.86i  LICENSE
Cacti-plugin-0.8.6i.diff  files-0.86i  Readme.txt
[root@lib cacti-plugin-arch]# ▌
```

（2）有两种方法来安装 Cacti 的插件架构扩展。第一种方法是使用 patch 文件。patch 文件包含了原始文件与修改后文件的不同之处，所以可以使用 patch 文件来得到新的文件。第二种方法是使用 pre-patched 文件进行直接覆盖，将与 Cacti 相对应版本的 files-0.8.6*目录下的文件直接复制到 Cacti 目录下进行覆盖。在此选择使用 patch 文件进行安装（这也是官网推荐的方法）。将与 Cacti 相应版本的 cacti-pligin-0.8.6*.diff 文件复制到 Cacti 目录下，然后使用 patch 命令进行安装。

```
# cp cacti-plugin-0.8.6j.diff /usr/local/apache/htdocs/cacti
//备份 Cacti 目录，以备插件扩展安装失败后能恢复到原状态
# cd /usr/local/apache/htdocs
# cp -r cacti ./cacti.bak
```

```
# cd ./cacti
//首先使用以下命令进行测试
# patch -p1 -N --dry-run < cacti-plugin-0.8.6j.diff
//以上命令成功后，使用以下命令进行安装
# patch -p1 -N < cacti-plugin-0.8.6j.diff
```

（3）安装后配置。首先查看你的 Cacti 配置文件，看 Cacti 相关的数据库信息是否被覆盖，如果被覆盖，请直接从备份中复制一份到配置文件目录。

```
# cp../include/config.php ./include/config.php
打开Cacti配置文件，找到以下选项
```

```
$config['url_path'] = "/";
```

如果你的 Cacti 能够在浏览器中使用以下方法直接访问，则不用修改 Cacti 配置文件。

```
http://www.youdomain.com 或
http://Cacti机器IP
```

如果 Cacti 在浏览器中使用以下方法直接访问：

```
http://www.youdomain.com/cacti
或http://cacti机器ip/cacti
```

则“$config['url_path'] = "/"”；必须被修改为“$config['url_path'] = "/cacti/"”。

以上需要注意前后的“/”不能丢。至此，Cacti 插件结构的扩展完成，现在可以下载并安装你需要的 Cacti 插件了。

2．安装插件

插件的安装、升级、移除是很容易的，在安装插件前你必须确保安装了 Cacti 插件结构扩展。

（1）下载插件。在此以 Monitor 插件为例。Monitor 插件下载地址：http://cactiusers.org//downloads 执行命令解压下载的 tar 包，解压后你会得到一个 monitor 目录。

```
# tar -zvcf monitor-0.7.tar.gz
```

（2）安装 Monitor。只需要将 monitor 目录拷贝到 cacti/plugins/目录下即可。如果是升级安装，只需要完全覆盖原 monitor 目录即可。

（3）激活 monitor 插件。修改 Cacti 配置文件（Cacti 目录下的 include/config.php）。在配置中查找“$plugins = array();”行，在此行下面加入：

```
$plugins[] = 'monitor';
```

**注意** 上面 monitor 的名字必须与 cacti/plugins/目录下插件目录的名字相同。

（4）有些插件需要进行额外的配置，请根据插件的安装文档进行配置，在此略述。

（5）移除插件时只要修改 Cacti 配置文件，注释掉与插件相关的行即可。

例如：

```
//$plugins[] = 'monitor';
```

（6）在 console 选项卡，单击左侧菜单中的 Settings 链接，在右侧出现的 Web 选项界面中切

换到 Misc 选项卡来配置 Monitor 插件，如图 16-31 所示。

General | Paths | Poller | Graph Export | Visual | Authentication | Alerting/Thold | Misc

**Cacti Settings (Misc)**

**Monitor**

| Setting | Value |
| --- | --- |
| **Alarm Sound**<br>This is the sound file that will be played when a host is down. | attn-noc.wav |
| **Refresh Interval**<br>This is the time in seconds before the page refreshes. (1 - 300) | 300 |
| **Icon Spacing**<br>This is how many icons to show per line. (1 - 20) | 10 |
| **Show Icon Legend**<br>Check this to show an icon legend on the Monitor display | ☐ Show Icon Legend |

图 16-31　Misc 选项卡

以下所看到的是 Monitor 选项界面下所监控的机器状态。当有机器 Down 掉后，相应机器图标将由绿色变成红色，并发出声音进行报警，如图 16-32 所示。

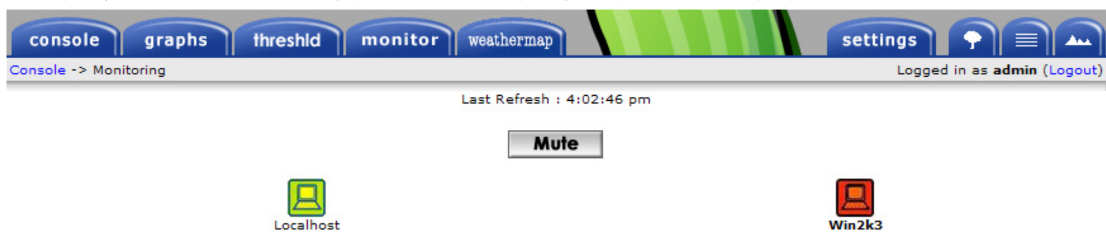



图 16-32　设置插件

# 第 17 章

# 他山之石：发现企业网络漏洞

## 本章导读

网络攻防其实是一场非常激烈的“博弈”。攻防双方总是在不断地寻找对方的漏洞，发动攻击，再发现对方漏洞，再次发动攻击的过程中。而网络安全工作者最可贵的一点应该是需要具备“换位思考”的能力。所谓换位思考，就是在作网络安全防护的时候，应该时时想到对方会如何来对自己的网络进行攻击，会采取什么样的技术和手段，而自己又存在什么样的漏洞和弱点被对方利用？做到这一点，就犹如三国中的诸葛亮，做到用兵如神，也就能从容地应对网络攻击了。

在网络高速发展的今天，目前的攻击方总是首先会通过端口扫描和漏洞扫描的方法来获得被攻击方的开放服务及其漏洞信息，再通过黑客工具来发动后续攻击。那么只要我们 500 强企业能够先于攻击者对企业网络的漏洞进行扫描，提前做好加固和防范工作，也就能够做好网络安全防护的第一项工作。因此，本章将介绍如何在 Linux 系统下通过端口和漏洞扫描来发现企业网络漏洞，从而给安全工作者提供相应的素材，以针对具体信息采取相应的措施来对该漏洞进行修补等。

## 17.1 发现企业网络漏洞的大致思路

### 17.1.1 基本思路

随着 Internet 的不断发展，信息技术已成为促进经济发展、社会进步的巨大推动力：当今社会高度的计算机化信息资源对任何人无论在任何时候、任何地方都变得极有价值。不管是存储在工作站中、服务器里还是流通于 Internet 上的信息都已转变成为一个关系事业成败关键的策略点，这就使保证信息的安全变得格外重要。

安全扫描技术是一类重要的网络安全技术。安全扫描技术与防火墙、入侵检测系统互相配合，能够有效提高网络的安全性。通过对网络的扫描，网络管理员可以了解网络的安全配置和运行的应用服务，及时发现安全漏洞，客观评估网络风险等级。网络管理员可以根据扫描的结果更正网络安全漏洞和系统中的错误配置，在黑客攻击前进行防范。如果说防火墙和网络监控系统是被动的防御手段，那么安全扫描就是一种主动的防范措施，可以有效地避免黑客地攻击行为，做到防患于未然。

安全扫描技术主要分为两类：主机安全扫描技术和网络安全扫描技术。网络安全扫描技术主要针对系统中不合适的设置脆弱的口令，以及针对其他同安全规则抵触的对象进行检查等；而主机安全扫描技术则是通过执行一些脚本文件模拟对系统进行攻击的行为并记录系统的反应，从而发现其中的漏洞。

由于公司网络使用已经非常普遍，能够通过网络进行扫描是最简单不过了，既安全又方便，且网络扫描的技术和工具都已经非常成熟，因此决定采用网络安全扫描。

### 17.1.2 采用网络安全扫描

网络安全扫描技术是一种基于 Internet 远程检测目标网络或本地主机安全性脆弱点的技术。通过网络安全扫描，系统管理员能够发现所维护的 Web 服务器的各种 TCP/IP 端口的分配、开放的服务、Web 服务软件版本和这些服务及软件呈现在 Internet 上的安全漏洞。网络安全扫描技术也是采用积极的、非破坏性的办法来检验系统是否有可能被攻击崩溃。它利用了一系列的脚本模拟对系统进行攻击的行为，并对结果进行分析。这种技术通常被用来进行模拟攻击实验和安全审计。网络安全扫描技术与防火墙、安全监控系统互相配合就能够为网络提供很高的安全性。

一次完整的网络安全扫描分为以下三个阶段。

（1）发现目标主机或网络。

（2）发现目标后进一步搜集目标信息，包括操作系统类型、运行的服务以及服务软件的版本等。如果目标是一个网络，还可以进一步发现该网络的拓扑结构、路由设备以及各主机的信息。

（3）根据搜集到的信息判断或者进一步测试系统是否存在安全漏洞。

网络安全扫描技术包括 PING 扫射（Ping sweep）、操作系统探测（Operating system identification）、如何探测访问控制规则（firewalking）、端口扫描（Port scan）以及漏洞扫描（vulnerability scan）等。这些技术在网络安全扫描的三个阶段中各有体现。PING 扫射用于网络安全扫描的第 1 阶段，可以帮助我们识别系统是否处于活动状态。操作系统探测、如何探测访问控制规则和端口扫描用于网络安全扫描的第 2 阶段，其中操作系统探测顾名思义就是对目标主机

运行的操作系统进行识别；如何探测访问控制规则用于获取被防火墙保护的远端网络的资料；而端口扫描是通过与目标系统的 TCP/IP 端口连接，并查看该系统处于监听或运行状态的服务。网络安全扫描第 3 阶段采用的漏洞扫描通常是在端口扫描的基础上，对得到的信息进行相关处理，进而检测出目标系统存在的安全漏洞。

端口扫描技术和漏洞扫描技术是网络安全扫描技术中的两种核心技术，并且广泛运用于当前较成熟的网络扫描器中，如著名的 Nmap 和 Nessus。鉴于这两种技术在网络安全扫描技术中起着举足轻重的作用，下面对这两种技术进行了非常系统和完整的学习、总结和研究。

## 17.2 端口扫描

### 17.2.1 端口扫描技术的基本原理

对于位于网络中的计算机系统来说，一个端口就是一个潜在的通信通道，也就是一个入侵通道。对目标计算机进行端口扫描，能得到许多有用的信息，从而发现系统的安全漏洞。通过其可以使系统用户了解系统目前向外界提供了哪些服务，从而为系统用户管理网络提供了一种参考的手段。

从技术原理上来说，端口扫描向目标主机的 TCP/UDP 服务端口发送探测数据包，并记录目标主机的响应。通过分析响应来判断服务端口是打开还是关闭，就可以得知端口提供的服务或信息。端口扫描也可以通过捕获本地主机或服务器的流入、流出 IP 数据包来监视本地主机的运行情况，不仅能对接收到的数据进行分析，而且能够帮助用户发现目标主机的某些内在的弱点，而不会提供进入一个系统的详细步骤。一般来说，端口扫描的目的通常是以下的一项或者多项。

- 发现开放端口：发现目标系统上开放的 TCP 或 UDP 端口。
- 了解主机操作系统信息：端口扫描可以通过操作系统的“指纹”来推测被扫描操作系统或者应用程序的版本等信息。
- 了解软件或者服务版本：软件或服务版本可以通过标志获取或者应用程序的指纹来识别获得。
- 发现脆弱的软件版本：识别软件和服务的缺陷，从而有助于发起针对漏洞的攻击。

端口扫描主要有经典的（全连接）扫描器以及所谓的 SYN（半连接）扫描器，此外还有间接扫描和秘密扫描等。TCP 扫描方式是通过与被扫描主机建立标准的 TCP 连接，因此这种方式最准确，很少漏报、误报，但是容易被目标主机察觉、记录。SYN 方式是通过与目标主机建立半打开连接，这样就不容易被目标主机记录，但是扫描结果会出现漏报，在网络状况不好的情况下这种漏报是严重的。

### 17.2.2 端口扫描技术的主要种类

#### 1. TCP 全连接扫描

全连接扫描是 TCP 端口扫描的基础，现有的全连接扫描有 TCP connect()扫描和 TCP 反向 ident 扫描等。

其中 TCP connect()扫描的实现原理如下：扫描主机通过 TCP/IP 协议的三次握手与目标主机的指定端口建立一次完整的连接。连接由系统调用 connect 开始。如果端口开放，则连接将建立成功；否则，若返回-1，则表示端口关闭。如果建立连接成功，则响应扫描主机的 SYN/ACK 连接请求，这一响应表明目标端口处于监听（打开）的状态。如果目标端口处于关闭状态，则目标主机会向扫描主机发送 RST 的响应。

而反向 ident 扫描协议允许看到通过 TCP 连接的任何进程的拥有者的用户名，即使这个连接的不是由这个进程开始的。比如，连接到 http 端口，然后用 identd 来发现服务器是否正在以 root 权限运行。这种方法的明显缺点是只能在和目标端口建立了一个完整的 TCP 连接后才能看到。

### 2．TCP 半连接（SYN）扫描

若端口扫描没有完成一个完整的 TCP 连接，在扫描主机和目标主机的一指定端口建立连接的时候只完成了前两次握手，在第三步时，扫描主机中断了本次连接，使连接没有完全建立起来，这样的端口扫描称为半连接扫描，也称为间接扫描。现有的半连接扫描有 TCP SYN 扫描和 IP ID 头 dumb 扫描等。

SYN 扫描的优点在于即使日志中对扫描有所记录，但是尝试进行连接的记录也要比全扫描少得多。缺点是在大部分操作系统下，发送主机需要构造适用于这种扫描的 IP 包，通常情况下，构造 SYN 数据包需要超级用户或者授权用户访问专门的系统调用。

### 3．UDP 扫描

UDP 扫描中不使用工具来设定特定的状态标志。如果源 UDP 包的响应（即 ICMP 端口不可达消息）说明端口已经“关闭”，UDP 扫描的过程比较长，速度比较慢。

### 4．标志获取扫描

标志获取扫描是指连接到系统的特定端口，检查监听该端口的应用软件标志的过程。它可以利用 TCP 端口连接来获得特定系统上运行的软件和程序版本信息。在实际的应用过程中，系统管理员可能会更改或者删除相关标志，从而隐藏被监听的应用程序。

### 5．包分片

许多端口扫描器都支持包分片功能。这个功能有助于包穿透包过滤设备并逃避入侵检测系统的监测。包分片技术将 TCP 或者 UDP 包头分成多个包，增加访问控制设备检测端口扫描信息的难度。当前大多数防火墙和 IDS 实现都能够在评估原始 IP 包前先对其进行重组，这样可以阻挡包分片的企图。但一些旧的防火墙和 IDS 设备缺乏这种功能。

### 6．欺骗扫描

Nmap 和其他一些端口扫描工具都有“欺骗”功能，能够在直接扫描的同时进行一个或者多个欺骗性的扫描。由于这种欺骗性通常掩盖了真实源地址，并与真实的扫描同时进行，所以目标系统追踪扫描来源的难度大大增加。

### 7．标识扫描

标识扫描能够用来识别与特定 TCP 连接绑定的用户账户，它可以通过与 113 号 TCP 端口进行通信来完成，该端口会返回该连接所有者的身份信息。这种扫描仅仅对于运行 ident 服务的系

统有效，也可以用于识别使用特权账号（例如root）的服务。

### 8. FTP反弹扫描

FTP反弹扫描利用FTP服务器进行欺骗扫描，它利用的是FTP协议对代理FTP连接的支持这一特性。利用FTP服务器作为反弹“代理”，黑客能够隐藏源扫描器的原始地址。

### 9. 源端口扫描

端口扫描工具中的源端口扫描允许扫描者设置静态的TCP或者UDP源扫描端口，以避开包过滤访问控制设备。用于扫描的源端口通常与常用服务的端口相关联（比如HTTP、DNS、SMTP、FTP等），这些端口经常处于访问控制设备的许可范围之内。

### 10. 主机扫描

主机扫描的目的是确定在目标网络上的主机是否可达。这是信息收集的初级阶段，其效果直接影响到后续的扫描。它主要包括以下几类。

- ICMP Echo扫描。
- ICMP Sweep扫描。
- Broadcast ICMP扫描。
- Non-Echo ICMP扫描。

1）ICMP Echo和ICMP Sweep扫描

Ping的实现机制，判断在一个网络上主机是否开机时非常有用。向目标主机发送ICMP Echo Request （type 8）数据包，等待回复的ICMP Echo Reply（type 0）数据包。如果能收到，则表明目标系统可达，否则表明目标系统不可达或发送的包被对方的设备过滤掉。该机制的优点是简单、系统支持；缺点是很容易被防火墙限制。在实际使用过程中，可以通过并行发送，同时探测多个目标主机，以提高探测效率（ICMP Sweep扫描）。

2）Broadcast ICMP扫描

该扫描机制将ICMP请求包的目标地址设为广播地址或网络地址，则可以探测广播域或整个网络范围内的主机。该机制明显的缺点是只适合于UNIX/Linux系统，Windows操作系统会忽略这种请求包。并且，这种扫描方式容易引起广播风暴，从而占用大量的网络资源。

3）Non-Echo ICMP扫描

该扫描机制指的是除上述3种之外的一些其他ICMP类型包，它们也可以用于对主机或网络设备的探测，例如：

- Stamp Request（Type 13）
- Reply（Type 14）
- Information Request（Type 15）
- Reply（Type 16）
- Address Mask Request（Type 17）
- Reply（Type 18）

### 11. 操作系统“指纹”扫描

根据各个操作系统在TCP/IP协议栈实现上的不同特点，采用黑盒测试方法，通过研究其对

各种探测的响应形成识别指纹（footprint），进而识别目标主机运行的操作系统。根据采集指纹信息的方式，又可以分为被动扫描和主动扫描两种方式。

1）被动扫描

通过抓包程序（TCPDUMP、Sniffer、Wireshark 等）收集数据包，再对数据包的不同特征（TCP Window Size、IP TTL、IP TOS、DF 位等参数）进行分析，来识别操作系统。被动扫描基本不具备攻击特征，具有很好的隐蔽性，但其实现严格依赖扫描主机所处的网络拓扑结构；和主动探测相比较，具有速度慢、可靠性不高等缺点。

2）主动扫描

采用向目标系统发送构造的特殊包并监控其应答的方式来识别操作系统类型。主动扫描具有速度快、可靠性高等优点，但同样严重依赖于目标系统网络拓扑结构和过滤规则。

主动扫描主要包括以下几种技术。

- FIN 探测：发送一个 FIN 包给一个打开的端口，一般的行为是不响应。
- BOGUS 标记探测：设置一个未定义的 TCP“标记”（64 或 128）在 SYN 包的 TCP 头里。Linux 系统到内核 2.0.35 版本之前在回应中保持该标记。
- TCP ISN 取样：其原理是通过在操作系统对连接请求的回应中寻找 TCP 连接初始化序列号的特征。
- 不分段位：许多操作系统开始在送出的一些包中设置 IP 的“Don't Fragment”（不分段）位。
- TCP 初始化窗口：检查返回包的窗口大小。如 Queso 和 Nmap 保持对窗口的精确跟踪，因为它对于特定 OS 基本是常数。
- ACK 值：不同实现中一些情况下 ACK 域的值是不同的。如果发送了一个 FIN|PSH|URG 到一个关闭的 TCP 端口。大多数实现会设置 ACK 为发送的初始序列数，而 Windows 会回送序列数加 1。
- ICMP 错误信息终结：一些操作系统跟从限制各种错误信息的发送率。例如，Linux 内核限制目的不可达消息的生成每 4 秒钟 80 个。测试的一种办法是发一串包到一些随机的高 UDP 端口并计数收到的不可达消息。
- ICMP 消息引用：ICMP 错误消息可以引用一部分引起错误的源消息。对一个端口不可达消息，几乎所有实现只送回 IP 请求头外加 8 个字节。然而，Solaris 送回的稍多，而 Linux 更多。
- SYN 洪水限度：如果收到过多的伪造 SYN 数据包，一些操作系统会停止新的连接尝试。许多操作系统只能处理 8 个包。

### 17.2.3 快速安装 Nmap

#### 1. Nmap 简介

Nmap 是一个网络探测和安全扫描程序，系统管理者和个人可以使用该软件扫描大型的网络，获取哪台主机正在运行以及提供什么服务等信息。Nmap 支持很多扫描技术，例如：UDP、TCP connect()、TCP SYN（半开扫描）、ftp 代理（bounce 攻击）、反向标志、ICMP、FIN、ACK 扫描、圣诞树（Xmas Tree）、SYN 扫描和 null 扫描。从扫描类型小节中可以得到细节。Nmap 还提供了一些高级的特征，例如：通过 TCP/IP 协议栈特征探测操作系统类型，秘密扫描，动态

延时和重传计算，并行扫描，通过并行 ping 扫描探测关闭的主机，诱饵扫描，避开端口过滤检测，直接 RPC 扫描（无须端口映射），碎片扫描，以及灵活的目标和端口设定。

为了提高 Nmap 在 non-root 状态下的性能，软件的设计者付出了很大的努力。很不幸，一些内核界面（例如 raw socket）需要在 root 状态下使用。所以应该尽可能在 root 状态下使用 Nmap。

Nmap 运行通常会得到被扫描主机端口的列表。Nmap 总会给出 well known 端口的服务名（如果可能）、端口号、状态和协议等信息。每个端口的状态有：open、filtered、unfiltered。

- open 状态意味着目标主机能够在这个端口使用 accept()系统调用接收连接。
- filtered 状态表示防火墙、包过滤和其他的网络安全软件掩盖了这个端口，禁止 Nmap 探测其是否打开。
- unfiltered 表示这个端口关闭，并且没有防火墙/包过滤软件来隔离 Nmap 的探测企图。通常情况下，端口的状态基本都是 unfiltered 状态，只有在大多数被扫描的端口处于 filtered 状态下，才会显示处于 unfiltered 状态的端口。

根据使用的功能选项，Nmap 也可以报告远程主机的下列特征：使用的操作系统、TCP 序列、运行绑定到每个端口上的应用程序的用户名、DNS 名、主机地址是否是欺骗地址，以及其他一些东西。

2．快速安装 Nmap

在使用之前，我们需要下载 Nmap 软件的源码包进行安装。目前，笔者下载到的最新版本为：nmap-5.00.tgz，下载网址为：http://linux.softpedia.com/get/System/Networking/Nmap-184.shtml。下载完成后，用户执行以下安装命令即可。

（1）解压缩软件包：

```
#tar -xzvf nmap-5.00.tgz
```

（2）切换到安装目录：

```
#cd nmap-5.00
```

（3）使用 configure 命令生成 make 文件：

```
#./configure
```

（4）编译源代码：

```
#make
```

（5）安装相关模块：

```
#make install
```

### 17.2.4　使用 Nmap 确定开放端口

功能选项可以组合使用。一些功能选项只能够在某种扫描模式下使用。Nmap 会自动识别无效或者不支持的功能选项组合，并向用户发出警告信息。使用 Nmap，可以参考使用以下一些常用的选项。

### 1．主要扫描选项

1）-sT

TCP connect()扫描：这是最基本的 TCP 扫描方式。connect()是一种系统调用，由操作系统提供，用来打开一个连接。如果目标端口有程序监听，connect()就会成功返回，否则这个端口是不可达的。这项技术最大的优点是，用户不需要 root 权限，任何 Linux 用户都可以自由地使用这个系统调用。这种扫描很容易被检测到，在目标主机的日志中会记录大批的连接请求以及错误信息。

2）-sS

TCP 同步扫描（TCP SYN）：因为不必全部打开一个 TCP 连接，所以这项技术通常称为半开扫描（half-open）。用户可以发出一个 TCP 同步包（SYN），然后等待回应。如果对方返回 SYN|ACK（响应）包就表示目标端口正在监听；如果返回 RST 数据包，则表示目标端口没有监听程序；如果收到一个 SYN|ACK 包，源主机就会马上发出一个 RST（复位）数据包断开和目标主机的连接，这实际上是由我们的操作系统内核自动完成的。这项技术最大的好处是，很少有系统能够把它记入系统日志。不过，用户需要 root 权限来定制 SYN 数据包。

3）-sF -sX -sN

秘密 FIN 数据包扫描、圣诞树（Xmas Tree）、空（Null）扫描模式：即使 SYN 扫描都无法确定的情况下使用。一些防火墙和包过滤软件能够对发送到被限制端口的 SYN 数据包进行监视，而且有些程序比如 synlogger 和 courtney 能够检测那些扫描。这些高级的扫描方式可以逃过这些干扰。这些扫描方式的理论依据是：关闭的端口需要对用户的探测包回应 RST 包，而打开的端口必须忽略有问题的包。FIN 扫描使用暴露的 FIN 数据包来探测，而圣诞树扫描打开数据包的 FIN、URG 和 PUSH 标志。不幸的是，微软决定完全忽略这个标准，另起炉灶。所以这种扫描方式对 Windows 95/NT 无效。不过，从另外的角度讲，可以使用这种方式来分辨两种不同的平台。如果使用这种扫描方式可以发现打开的端口，用户就可以确定目标主机运行的不是 Windows 系统。如果使用-sF、-sX 或者-sN 扫描显示所有的端口都是关闭的，而使用 SYN 扫描显示有打开的端口，用户可以确定目标主机运行可能的是 Windows 系统。现在这种方式没有什么太大的用处，因为 Nmap 有内嵌的操作系统检测功能。还有其他几个系统使用和 Windows 同样的处理方式，包括 Cisco、BSDI、HP/UX、MYS、IRIX。在应该抛弃数据包时，以上这些系统都会从打开的端口发出复位数据包。

4）-sP

ping 扫描：有时用户只是想知道此时网络上哪些主机正在运行。通过向用户指定的网络内的每个 IP 地址发送 ICMP echo 请求数据包，Nmap 就可以完成这项任务。如果主机正在运行就会作出响应。不幸的是，一些站点会主动阻塞 ICMP echo 请求数据包。然而，在默认的情况下 Nmap 也能够向 80 端口发送 TCP ack 包，如果用户收到一个 RST 包，则表示主机正在运行。Nmap 使用的第三种技术是：发送一个 SYN 包，然后等待一个 RST 或者 SYN/ACK 包。对于非 root 用户，Nmap 使用 connect()方法。在默认的情况下（root 用户），Nmap 并行使用 ICMP 和 ACK 技术。值得注意的是：Nmap 在任何情况下都会进行 ping 扫描，只有目标主机处于运行状态，才会进行后续的扫描。如果用户只是想知道目标主机是否运行，而不想进行其他扫描，才会用到这个选项。

5）-sU

UDP 扫描：如果用户想知道在某台主机上提供哪些 UDP 服务，可以使用这种扫描方法。Nmap 首先向目标主机的每个端口发出一个 0 字节的 UDP 包，如果我们收到端口不可达的 ICMP 消息，

端口就是关闭的，否则我们就假设它是打开的。

6）-sA

ACK 扫描：这项高级的扫描方法通常用来穿过防火墙的规则集。通常情况下，这有助于确定一个防火墙是功能比较完善的或者是一个简单的包过滤程序，只是阻塞进入的 SYN 包。这种扫描是向特定的端口发送 ACK 包（使用随机的应答/序列号）。如果返回一个 RST 包，这个端口就标记为 unfiltered 状态。如果什么都没有返回，或者返回一个不可达 ICMP 消息，这个端口就归入 filtered 类。注意，Nmap 通常不输出 unfiltered 的端口，所以在输出中通常不显示所有被探测的端口。显然，这种扫描方式不能找出处于打开状态的端口。

7）-sW

对滑动窗口的扫描：这项高级扫描技术非常类似于 ACK 扫描，除了它有时可以检测到处于打开状态的端口，因为滑动窗口的大小是不规则的，有些操作系统可以报告其大小。这些系统至少包括：某些版本的 AIX、Amiga、BeOS、BSDI、Cray、Tru64 UNIX、DG/UX、OpenVMS、Digital UNIX、OpenBSD、OpenStep、QNX、Rhapsody、SunOS 4.x、Ultrix、VAX、VXWORKS。从 nmap-hackers 邮件 3 列表的文档中可以得到完整的列表。

8）-sR

RPC 扫描：这种方法和 Nmap 的其他不同的端口扫描方法结合使用。选择所有处于打开状态的端口向它们发出 SunRPC 程序的 NULL 命令，以确定它们是否是 RPC 端口，如果是，就确定是哪种软件及其版本号。因此用户能够获得防火墙的一些信息。诱饵扫描目前还不能和 RPC 扫描结合使用。

### 2．通用扫描选项

1）-P0

在扫描之前，不必 ping 主机。有些网络的防火墙不允许 ICMP echo 请求穿过，使用该选项可以对这些网络进行扫描。microsoft.com 就是一个例子，因此在扫描该站点时，用户应该一直使用-P0 或者-PT 80 选项。

2）-PT

扫描之前，使用 TCP ping 确定哪些主机正在运行。Nmap 不是通过发送 ICMP echo 请求包然后等待响应来实现这种功能，而是向目标网络（或者单一主机）发出 TCP ACK 包然后等待回应。如果主机正在运行就会返回 RST 包。只有在目标网络/主机阻塞了 ping 包，而仍旧允许用户对其进行扫描时，该选项才有效。对于非 root 用户，我们使用 connect()系统调用来实现这项功能。使用-PT <端口号>来设定目标端口。默认的端口号是 80，因为这个端口通常不会被过滤。

3）-PS

对于 root 用户，该选项让 Nmap 使用 SYN 包而不是 ACK 包来对目标主机进行扫描。如果主机正在运行，则返回一个 RST 包（或者一个 SYN/ACK 包）。

4）-PI

设置该选项，让 Nmap 使用真正的 ping（ICMP echo 请求）来扫描目标主机是否正在运行。使用该选项让 Nmap 发现正在运行的主机的同时，Nmap 也会对用户的直接子网广播地址进行观察。直接子网广播地址，把外部的包转换为一个内向的 IP 广播包，向一个计算机子网发送。这些 IP 广播包应该删除，因为会造成拒绝服务攻击（例如 smurf）。

5）-PB

这是默认的 ping 扫描选项。它使用 ACK（-PT）和 ICMP（-PI）两种扫描类型并行扫描。如果防火墙能够过滤其中一种包，使用这种方法，用户就能够穿过防火墙。

6）-O

该选项激活对 TCP/IP 指纹特征（fingerprinting）的扫描，获得远程主机的标志。换句话说，Nmap 使用一些技术检测目标主机操作系统网络协议栈的特征。Nmap 使用这些信息建立远程主机的指纹特征，把它和已知的操作系统指纹特征数据库做比较，就可以知道目标主机操作系统的类型。

7）-I

该选项打开 Nmap 的反向标志扫描功能。Dave Goldsmith 1996 年向 bugtap 发出的邮件注意到这个协议，ident 协议允许使用 TCP 连接给出任何进程拥有者的用户名，即使这个进程并没有初始化连接。例如，用户可以连接到 HTTP 端口，接着使用 identd 确定这个服务器是否由 root 用户运行。这种扫描只能在同目标端口建立完全的 TCP 连接时（例如：-sT 扫描选项）才能成功。使用-I 选项时，远程主机的 identd 精灵进程就会查询在每个打开的端口上监听的进程的拥有者。显然，如果远程主机没有运行 identd 程序，这种扫描方法无效。

8）-f

该选项使 Nmap 使用碎片 IP 数据包发送 SYN、FIN、XMAS、NULL。使用碎片数据包增加包过滤、入侵检测系统的难度，使其无法知道用户的企图。因此，在 Nmap 中使用了 24 个字节的碎片数据包。虽然包过滤器和防火墙不能防这种方法，但是有很多网络出于性能上的考虑，禁止数据包的分片。注意该选项不能在所有的平台上使用。它在 Linux、FreeBSD、OpenBSD 以及其他一些 UNIX 系统下能够很好地工作。

9）-v

冗余模式。强烈推荐使用该选项，它会给出扫描过程中的详细信息。使用这个选项，用户可以得到事半功倍的效果。使用-d 选项可以得到更加详细的信息。

10）-h

快速参考选项。

11）oN

把扫描结果重定向到一个可读的文件 logfilename 中。

12）-oM

把扫描结果重定向到 logfilename 文件中，这个文件使用主机可以解析的语法。用户可以使用-oM 来代替 logfilename，这样输出就被重定向到标准输出 stdout。在这种情况下，正常的输出将被覆盖，错误信息仍然可以输出标准错误 stderr。要注意，如果同时使用了-v 选项，在屏幕上会打印出其他的信息。

13）-resume

某个网络扫描可能由于 control-C 或者网络损失等原因被中断，使用该选项可以使扫描接着以前的扫描进行。logfilename 是被取消扫描的日志文件，它必须是可读形式或者机器可以解析的形式。而且接着进行的扫描不能增加新的选项，只能使用与被中断的扫描相同的选项。Nmap 会接着日志文件中的最后一次成功扫描进行新的扫描。

14）-iL

从 inputfilename 文件中读取扫描的目标。在这个文件中要有一个主机或者网络的列表，由空格键、制表键或者回车键作为分隔符。如果使用-iL -，Nmap 就会从标准输入 stdin 读取主机的名字。用户可以从指定目标小节得到更加详细的信息。

15）-iR

让 Nmap 自己随机挑选主机进行扫描。

16）-p <端口范围>

该选项让用户选择要进行扫描的端口号的范围。例如，-p 23 表示：只扫描目标主机的 23 号端口。-p 20-30、139、60000-表示：扫描 20～30 号端口，139 号端口以及所有大于 60000 的端口。在默认情况下，Nmap 扫描从 1～1024 号以及 nmap-services 文件（如果使用 RPM 软件包，一般在/usr/share/nmap/目录中）中定义的端口列表。

17）-F

快速扫描模式，只扫描在 nmap-services 文件中列出的端口。显然比扫描所有的 65535 个端口要快。

18）-D

使用诱饵扫描方法对目标网络/主机进行扫描。如果 Nmap 使用这种方法对目标网络进行扫描，那么从目标主机/网络的角度来看，扫描就像从其他主机（decoy1，等）发出的。从而，即使目标主机的 IDS（入侵检测系统）对端口扫描发出报警，它们也不可能知道哪个是真正发起扫描的地址，哪个是无辜的。这种扫描方法可以有效地对付例如路由跟踪、response-dropping 等积极的防御机制，能够很好地隐藏用户的 IP 地址。每个诱饵主机名使用逗号分隔开，用户也可以使用 ME 选项，它代表用户自己的主机，和诱饵主机名混杂在一起。如果用户把 ME 放在第六或者更靠后的位置，一些端口扫描检测软件几乎根本不会显示用户的 IP 地址。如果用户不使用 ME 选项，Nmap 会把用户的 IP 地址随机夹杂在诱饵主机之中。需要注意的是：用户用来作为诱饵的主机应该正在运行或者用户只是偶尔向目标发送 SYN 数据包。很显然，如果在网络上只有一台主机运行，目标将很轻松地就能确定是哪台主机进行的扫描。或许，用户还要直接使用诱饵的 IP 地址而不是其域名，这样诱饵网络的域名服务器的日志上就不会留下关于用户的记录。还要注意：一些愚蠢的端口扫描检测软件会拒绝路由试图进行端口扫描的主机。因而，用户需要让目标主机和一些诱饵断开连接。如果诱饵是目标主机的网关或者就是其自己时，会给目标主机造成很大问题。所以用户需要慎重使用这个选项。诱饵扫描既可以在起始的 ping 扫描，也可以在真正的扫描状态下使用。它也可以和-O 选项组合使用。使用太多的诱饵扫描能够减缓用户的扫描速度甚至可能造成扫描结果不正确。同时，有些 ISP 会把用户的欺骗包过滤掉，仅管现在大多数的 ISP 不会对此进行限制。

19）-S <IP_Address>

在一些情况下，Nmap 可能无法确定用户的源地址。在这种情况下，可以使用该选项给出用户的 IP 地址。在欺骗扫描时，也使用这个选项。使用该选项可以让目标认为是其他的主机对自己进行扫描。

20）-e

告诉 Nmap 使用哪个接口发送和接收数据包。Nmap 能够自动对此接口进行检测，如果无效

就会告诉用户。

21）-g

设置扫描的源端口。一些简单的防火墙和包过滤器的规则集允许源端口为 DNS（53）或者 FTP-DATA（20）的包通过和实现连接。显然，如果攻击者把源端口修改为 20 或者 53，就可以摧毁防火墙的防护。在使用 UDP 扫描时，先使用 53 号端口；使用 TCP 扫描时，先使用 20 号端口。注意只有在能够使用这个端口进行扫描时，Nmap 才会使用该端口。例如，如果用户无法进行 TCP 扫描，Nmap 会自动改变源端口，即使用户使用了-g 选项。对于一些扫描，使用该选项会造成性能上的微小损失，因为有时会保存关于特定源端口的一些有用的信息。

22）-r

告诉 Nmap 不要打乱被扫描端口的顺序。

23）--randomize_hosts

使 Nmap 在扫描之前打乱每组扫描中的主机顺序，Nmap 每组可以扫描最多 2048 台主机。这样，可以使扫描更不容易被网络监视器发现，尤其和--scan_delay 选项组合使用，更能有效地避免被发现。

24）-M

设置进行 TCP connect()扫描时最多使用多少个套接字进行并行的扫描。使用该选项可以降低扫描速度，避免远程目标宕机。

### 3．高级设置选项

通常，Nmap 在运行时，能够很好地根据网络特点进行调整。扫描时，Nmap 会尽量减少被目标检测到的机会，同时尽可能加快扫描速度。然而 Nmap 默认的适时策略有时候不太适合用户的目标。使用下面这些选项，可以有效地控制 Nmap 的扫描时间。

1）-T

设置 Nmap 的扫描策略，主要有以下几种。

- Paranoid：为了避开 IDS 的检测使扫描速度极慢，Nmap 串行所有的扫描，至少每隔 5 分钟发送一个包。
- Sneaky：和 Paranoid 差不多，只是数据包的发送间隔是 15 秒。
- Polite：不增加太大的网络负载，避免宕掉目标主机，串行每个探测，并且使每个探测有 0.4 秒的间隔。
- Normal：Nmap 默认的选项，在不是网络过载或者主机/端口丢失的情况下尽可能快速地扫描。
- Aggressive：设置 5 分钟的超时限制，使对每台主机的扫描时间不超过 5 分钟，并且使对每次探测回应的等待时间不超过 1.5 秒。
- Insane：只适合快速的网络或者用户不在意丢失某些信息的情况，每台主机的超时限制是 75 秒，对每次探测只等待 0.3 秒。

用户也可以使用数字来代替上述这些模式，例如：-T 0 等于-T Paranoid，-T 5 等于-T Insane。只是上述这些模式不能和下面的选项组合使用。

2）--host_timeout

设置扫描一台主机的时间，以毫秒为单位。默认情况下没有超时限制。

3）--max_rtt_timeout

设置对每次探测的等待时间，以毫秒为单位。如果超过这个时间限制就重传或者超时。默认值大约是9000毫秒。

4）--min_rtt_timeout

当目标主机的响应很快时，Nmap就缩短每次探测的超时时间。这样会提高扫描的速度，但是可能会丢失某些响应时间比较长的包。使用该选项，可以让Nmap对每次探测至少等待用户指定的时间，以毫秒为单位。

5）--initial_rtt_timeout

设置初始探测的超时值。通常该选项只在使用-P0选项扫描有防火墙保护的主机时才有用。默认值是6000毫秒。

6）--max_parallelism

设置最大的并行扫描数量。--max_parallelism 1表示同时只扫描一个端口。该选项对其他的并行扫描也有效，例如ping sweep、RPC scan。

7）--scan_delay

设置在两次探测之间，Nmap必须等待的时间。该选项主要用于降低网络的负载。

#### 4．扫描实施第一步：发现活动主机

使用Nmap扫描整个网络寻找目标，以确定目标机是否处于连通状态。通过使用-sP命令，进行ping扫描。默认情况下，Nmap会给每个扫描到的主机发送一个ICMP echo和一个TCP ACK，主机对任何一种响应都会被Nmap得到，扫描速度非常快，在很短的时间内可以扫描一个很大的网络。该命令使用如下：

```
[root@localhost ~]# nmap -sP 10.1.4.0/24
Starting Nmap 5.0 ( http://www.insecure.org/nmap/ ) at 2009-12-17 15:09 CST
Host 10.1.4.1 appears to be up.
MAC Address: 00:0B:CD:B9:EE:A8 (Compaq (HP))
Host 10.1.4.6 appears to be up.
MAC Address: 00:1E:65:F0:48:B8 (Unknown)
Host 10.1.4.8 appears to be up.
MAC Address: 00:1F:3C:56:00:95 (Unknown)
Host 10.1.4.9 appears to be up.
MAC Address: 00:13:CE:FA:45:4A (Intel Corporate)
Host 10.1.4.14 appears to be up.
MAC Address: 00:21:5C:64:0D:07 (Unknown)
Host 10.1.4.15 appears to be up.
MAC Address: 00:21:5C:77:C1:83 (Unknown)
```

```
Host 10.1.4.16 appears to be up.
MAC Address: 00:1C:BF:A4:02:39 (Unknown)
Host 10.1.4.18 appears to be up.
MAC Address: 00:1C:BF:D5:30:04 (Unknown)
Host 10.1.4.20 appears to be up.
MAC Address: 00:1C:BF:95:6F:3C (Unknown)
Host 10.1.4.21 appears to be up.
MAC Address: 00:1C:BF:95:62:69 (Unknown)
Host 10.1.4.22 appears to be up.
MAC Address: 00:1F:3A:4C:90:D2 (Unknown)
Host 10.1.4.23 appears to be up.
Host 10.1.4.25 appears to be up.
MAC Address: 00:22:FA:CA:C6:7A (Unknown)
Host 10.1.4.26 appears to be up.
MAC Address: 00:26:BB:0F:40:01 (Unknown)
Host 10.1.4.27 appears to be up.
MAC Address: 00:22:FA:B0:BB:D2 (Unknown)
Host 10.1.4.28 appears to be up.
MAC Address: 00:1E:65:F2:76:96 (Unknown)
Host 10.1.4.29 appears to be up.
MAC Address: 00:1C:BF:9D:20:F5 (Unknown)
略。。。。。。
Nmap finished: 256 IP addresses (125 hosts up) scanned in 7.852 seconds
```

通过该扫描，可以发现该公司网络中有 125 台主机是活跃的，也就是说有机可趁，下一步就是要进行更详细的扫描，来扫描这些主机到底有些什么活动端口。

### 5．扫描实施第二步：扫描端口扫描

通常情况下，当 Nmap 的使用者确定了网络上运行的主机处于连通状态，下一步的工作就是进行端口扫描。端口扫描使用-sT 参数。如下面结果所示：

```
[root@localhost ~]# nmap -v  -sT 10.1.4.0/24
Starting Nmap 5.0 ( http://www.insecure.org/nmap/ ) at 2009-12-17 14:19 CST
Initiating ARP Ping Scan against 23 hosts [1 port/host] at 14:19
The ARP Ping Scan took 0.51s to scan 23 total hosts.
DNS resolution of 12 IPs took 0.29s.
Initiating SYN Stealth Scan against 5 hosts [1680 ports/host] at 14:19
Discovered open port 25/tcp on 10.1.4.1
Discovered open port 3389/tcp on 10.1.4.1
```

```
    Discovered open port 389/tcp on 10.1.4.1
    Discovered open port 21/tcp on 10.1.4.1
    Discovered open port 13/tcp on 10.1.4.1
    Discovered open port 80/tcp on 10.1.4.11
    Increasing send delay for 10.1.4.11 from 0 to 5 due to 11 out of 34 dropped
probes since last increase.
    Discovered open port 3372/tcp on 10.1.4.1
    Discovered open port 42/tcp on 10.1.4.1
    SYN Stealth Scan Timing: About 10.47% done; ETC: 14:24 (0:04:16 remaining)
    Discovered open port 143/tcp on 10.1.4.1
    Discovered open port 993/tcp on 10.1.4.1
    Discovered open port 1025/tcp on 10.1.4.1
    Discovered open port 445/tcp on 10.1.4.1
    Discovered open port 995/tcp on 10.1.4.1
    Increasing send delay for 10.1.4.11 from 5 to 10 due to max_successful_tryno
increase to 4
    Discovered open port 110/tcp on 10.1.4.1
    Discovered open port 17/tcp on 10.1.4.1
    Discovered open port 2008/tcp on 10.1.4.1
    Discovered open port 7/tcp on 10.1.4.1
    Discovered open port 9/tcp on 10.1.4.1
    Discovered open port 19/tcp on 10.1.4.1
    Increasing send delay for 10.1.4.11 from 10 to 20 due to max_successful_tryno
increase to 5
    Increasing send delay for 10.1.4.11 from 20 to 40 due to max_successful_tryno
increase to 6
    Discovered open port 6000/tcp on 10.1.4.1
    Discovered open port 135/tcp on 10.1.4.1
    Discovered open port 1029/tcp on 10.1.4.1
    Discovered open port 465/tcp on 10.1.4.1
    Discovered open port 139/tcp on 10.1.4.1
    Completed SYN Stealth Scan against 10.1.4.1 in 249.12s (4 hosts left)
    Completed SYN Stealth Scan against 10.1.4.6 in 257.58s (3 hosts left)
    Completed SYN Stealth Scan against 10.1.4.9 in 258.51s (2 hosts left)
    Completed SYN Stealth Scan against 10.1.4.11 in 274.71s (1 host left)
    Discovered open port 139/tcp on 10.1.4.10
```

```
The SYN Stealth Scan took 280.28s to scan 8400 total ports.
Host 10.1.4.1 appears to be up ... good.
Interesting ports on 10.1.4.1:
Not shown: 1657 closed ports
PORT     STATE SERVICE
7/tcp    open  echo
9/tcp    open  discard
13/tcp   open  daytime
17/tcp   open  qotd
19/tcp   open  chargen
21/tcp   open  ftp
25/tcp   open  smtp
42/tcp   open  nameserver
110/tcp  open  pop3
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
143/tcp  open  imap
389/tcp  open  ldap
445/tcp  open  microsoft-ds
465/tcp  open  smtps
993/tcp  open  imaps
995/tcp  open  pop3s
1025/tcp open  NFS-or-IIS
1029/tcp open  ms-lsa
2008/tcp open  conf
3372/tcp open  msdtc
3389/tcp open  ms-term-serv
6000/tcp open  X11
MAC Address: 00:0B:CD:B9:EE:A8 (Compaq (HP))

Host 10.1.4.6 appears to be up ... good.
All 1680 scanned ports on 10.1.4.6 are filtered
MAC Address: 00:1E:65:F0:48:B8 (Unknown)

Host 10.1.4.9 appears to be up ... good.
All 1680 scanned ports on 10.1.4.9 are filtered
```

```
MAC Address: 00:13:CE:FA:45:4A (Intel Corporate)

Host 10.1.4.10 appears to be up ... good.
Interesting ports on 10.1.4.10:
Not shown: 1678 filtered ports
PORT    STATE  SERVICE
139/tcp open   netbios-ssn
445/tcp closed microsoft-ds
MAC Address: 00:22:FA:DB:C3:A8 (Unknown)

Host 10.1.4.11 appears to be up ... good.
Interesting ports on 10.1.4.11:
Not shown: 1673 closed ports
PORT     STATE    SERVICE
80/tcp   open     http
170/tcp  filtered print-srv
179/tcp  filtered bgp
218/tcp  filtered mpp
736/tcp  filtered unknown
1720/tcp filtered H.323/Q.931
5302/tcp filtered hacl-cfg
MAC Address: 00:1E:65:F0:78:CA (Unknown)

DNS resolution of 1 IPs took 4.01s.
Initiating SYN Stealth Scan against 7 hosts [1680 ports/host] at 14:24
Increasing send delay for 10.1.4.20 from 0 to 5 due to 11 out of 11 dropped probes since last increase.
Increasing send delay for 10.1.4.20 from 5 to 10 due to 11 out of 11 dropped probes since last increase.
SYN Stealth Scan Timing: About 2.66% done; ETC: 14:43 (0:18:18 remaining)
Increasing send delay for 10.1.4.20 from 10 to 20 due to 11 out of 36 dropped probes since last increase.
SYN Stealth Scan Timing: About 7.69% done; ETC: 14:39 (0:13:50 remaining)
略。。。。。。
```

可以清楚地看到，端口扫描采用多种方式对网络中主机的 TCP 活动端口进行全面扫描，由于扫描的主机数太多（125 台），上面仅仅给出了两台主机的 TCP 端口情况，也就是主机 10.1.4.1

和 10.1.4.11，主机 10.1.4.1 打开的端口非常多，网络服务也相对比较丰富，并且从 IP 地址的构成来看，该主机极有可能是网关（一般网关的 IP 地址设定为 X.X.X.1），于是就锁定了这台主机进行后续的扫描。

### 6．扫描实施第三步：主机操作系统识别

通常一个入侵者可能对某个操作系统的漏洞很熟悉，能很轻易地进入此操作系统的机器。一个常见的选项是 TCP/IP 上的指纹，带有“-O”选项决定远程操作系统的类型。它可以和一个端口扫描结合使用，但不能和 ping 扫描结合使用。Nmap 通过向主机发送不同类型的探测信号，缩小查找的操作系统的范围。如下面的扫描结果所示：

```
[root@localhost ~]# nmap -O 10.1.4.1


Starting Nmap 5.0 ( http://www.insecure.org/nmap/ ) at 2009-12-17 15:28 CST
Interesting ports on 10.1.4.1:
Not shown: 1657 closed ports
PORT     STATE SERVICE
7/tcp    open  echo
9/tcp    open  discard
13/tcp   open  daytime
17/tcp   open  qotd
19/tcp   open  chargen
21/tcp   open  ftp
25/tcp   open  smtp
42/tcp   open  nameserver
110/tcp  open  pop3
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
143/tcp  open  imap
389/tcp  open  ldap
445/tcp  open  microsoft-ds
465/tcp  open  smtps
993/tcp  open  imaps
995/tcp  open  pop3s
1025/tcp open  NFS-or-IIS
1029/tcp open  ms-lsa
2008/tcp open  conf
3372/tcp open  msdtc
3389/tcp open  ms-term-serv
```

```
    6000/tcp open  X11
    MAC Address: 00:0B:CD:B9:EE:A8 (Compaq (HP))
    Device type: general purpose
    Running: Microsoft Windows 95/98/ME|NT/2K/XP
    OS details: Microsoft Windows Millennium Edition (Me), Windows 2000
Professional or Advanced Server, or Windows XP

    Nmap finished: 1 IP address (1 host up) scanned in 68.749 seconds
```

通过上述扫描结果可以看到，10.1.4.1 这台机器运行的是 Windows 系列的操作系统，并且提供了一些比较常用的网络服务。

#### 7．扫描实施第四步：扫描总结归纳

通过上述三步的扫描实施，逐步总结得出了以下几点结论和经验。

- 端口扫描的步骤是先发现活动主机，然后发现活动主机的活动端口，从而确定需要重点关注或者需要供给的主机，一般这些主机是服务端口开得多的主机；然后再对重点关注的主机实行操作系统扫描，从而确定操作系统类型，以便从操作系统层面来寻找突破口；一般情况下，如果是 Windows 操作系统的话，那么就有比较大的攻击或者是安全维护的空间。
- 除了上述几个常用的扫描选项外，在扫描过程中还需要综合采用其他-T 之类的选项，以保证扫描不被对方发现，否则对于具有丰富管理经验的管理员来说，很容易发现这种扫描行为，引起反追踪甚至是报复性的攻击等。当然，对于内网的网管性的维护工作来说，就没有这个风险。
- 在使用 Nmap 进行扫描后，对于它提供的开放端口号及其对应的服务名称，还可以使用表 17-1 进行比较对照，以更好地确认端口及其对应的开放服务，从而为下一步的针对性的攻击或者是网络管理作好准备。

表 17-1　常用端口及对应服务列表

| 端 口 号 | 端口服务名称 | 端口说明 |
|---|---|---|
| 1 | tcpmux | TCP 端口服务多路复用 |
| 5 | rje | 远程作业入口 |
| 7 | echo | echo 服务 |
| 9 | discard | 用于连接测试的空服务 |
| 11 | systat | 用于列举连接了的端口的系统状态 |
| 13 | daytime | 给请求主机发送日期和时间 |
| 17 | qotd | 给连接了的主机发送每日提示 |
| 18 | msp | 消息发送协议 |
| 19 | chargen | 字符生成服务；发送无止尽的字符流 |
| 20 | ftp-data | FTP 数据端口 |

续表

| 端 口 号 | 端口服务名称 | 端口说明 |
|---|---|---|
| 21 | ftp | 文件传输协议（FTP）端口；有时被文件服务协议（FSP）使用 |
| 22 | ssh | 安全 shell（SSH）服务 |
| 23 | telnet | Telnet 服务 |
| 25 | smtp | 简单邮件传输协议（SMTP） |
| 37 | time | 时间协议 |
| 39 | rlp | 资源定位协议 |
| 42 | nameserver | 互联网名称服务 |
| 43 | nicname | WHOIS 目录服务 |
| 49 | tacacs | 用于基于 TCP/IP 验证和访问的终端访问控制器访问控制系统 |
| 50 | re-mail-ck | 远程邮件检查协议 |
| 53 | domain | 域名服务（如 BIND） |
| 63 | whois++ | WHOIS++，被扩展了的 WHOIS 服务 |
| 67 | bootps | 引导协议（BOOTP）服务；还被动态主机配置协议（DHCP）服务使用 |
| 68 | bootpc | Bootstrap（BOOTP）客户；还被动态主机配置协议（DHCP）客户使用 |
| 69 | tftp | 小文件传输协议（TFTP） |
| 70 | gopher | gopher 互联网文档搜寻和检索 |
| 71 | netrjs-1 | 远程作业服务 |
| 72 | netrjs-2 | 远程作业服务 |
| 73 | netrjs-3 | 远程作业服务 |
| 73 | netrjs-4 | 远程作业服务 |
| 79 | finger | 用于用户联系信息的 finger 服务 |
| 80 | http | 用于万维网（WWW）服务的超文本传输协议（HTTP） |
| 88 | kerberos | Kerberos 网络验证系统 |
| 95 | supdup | Telnet 协议扩展 |
| 101 | hostname | SRI-NIC 机器上的主机名服务 |
| 102 | iso-tsap | ISO 开发环境（ISODE）网络应用 |
| 105 | csnet-ns | 邮箱名称服务器；也被 CSO 名称服务器使用 |
| 107 | rtelnet | 远程 Telnet |
| 109 | pop2 | 邮局协议版本 2 |
| 110 | pop3 | 邮局协议版本 3 |

续表

| 端　口　号 | 端口服务名称 | 端口说明 |
| --- | --- | --- |
| 111 | sunrpc | 用于远程命令执行的远程过程调用（RPC）协议，被网络文件系统（NFS）使用 |
| 113 | auth | 验证和身份识别协议 |
| 115 | sftp | 安全文件传输协议（SFTP）服务 |
| 117 | uucp-path | UNIX 到 UNIX 复制协议（UUCP）路径服务 |
| 119 | nntp | 用于 USENET 讨论系统的网络新闻传输协议（NNTP） |
| 123 | ntp | 网络时间协议（NTP） |
| 137 | netbios-ns | 在红帽企业 Linux 中被 Samba 使用的 NetBIOS 名称服务 |
| 138 | netbios-dgm | 在红帽企业 Linux 中被 Samba 使用的 NetBIOS 数据报服务 |
| 139 | netbios-ssn | 在红帽企业 Linux 中被 Samba 使用的 NetBIOS 会话服务 |
| 143 | imap | 互联网消息存取协议（IMAP） |
| 161 | snmp | 简单网络管理协议（SNMP） |
| 162 | snmptrap | SNMP 的陷阱 |
| 163 | cmip-man | 通用管理信息协议（CMIP） |
| 164 | cmip-agent | 通用管理信息协议（CMIP） |
| 174 | mailq | MAILQ |
| 177 | xdmcp | X 显示管理器控制协议 |
| 178 | nextstep | NextStep 窗口服务器 |
| 179 | bgp | 边界网络协议 |
| 191 | prospero | Cliffod Neuman 的 Prospero 服务 |
| 194 | irc | 互联网中继聊天（IRC） |
| 199 | smux | SNMP UNIX 多路复用 |
| 201 | at-rtmp | AppleTalk 选路 |
| 202 | at-nbp | AppleTalk 名称绑定 |
| 204 | at-echo | AppleTalk echo 服务 |
| 206 | at-zis | AppleTalk 区块信息 |
| 209 | qmtp | 快速邮件传输协议（QMTP） |
| 210 | z39.50 | NISO Z39.50 数据库 |
| 213 | ipx | 互联网络分组交换协议（IPX），被 Novell Netware 环境常用的数据报协议 |
| 220 | imap3 | 互联网消息存取协议版本 3 |
| 245 | link | LINK |

续表

| 端 口 号 | 端口服务名称 | 端口说明 |
|---|---|---|
| 347 | fatserv | Fatmen 服务器 |
| 363 | rsvp_tunnel | RSVP 隧道 |
| 369 | rpc2portmap | Coda 文件系统端口映射器 |
| 370 | codaauth2 | Coda 文件系统验证服务 |
| 372 | ulistproc | UNIX Listserv |
| 389 | ldap | 轻型目录存取协议（LDAP） |
| 427 | svrloc | 服务位置协议（SLP） |
| 434 | mobileip-agent | 可移互联网协议（IP）代理 |
| 435 | mobilip-mn | 可移互联网协议（IP）管理器 |
| 443 | https | 安全超文本传输协议（HTTP） |
| 444 | snpp | 小型网络分页协议 |
| 445 | microsoft-ds | 通过 TCP/IP 的服务器消息块（SMB） |
| 464 | kpasswd | Kerberos 口令和钥匙更换服务 |
| 468 | photuris | photuris 会话钥匙管理协议 |
| 487 | saft | 简单不对称文件传输（SAFT）协议 |
| 488 | gss-http | 用于 HTTP 的通用安全服务（GSS） |
| 496 | pim-rp-disc | 用于协议独立的多址传播（PIM）服务的会合点发现（RP-DISC） |
| 500 | isakmp | 互联网安全关联和钥匙管理协议（ISAKMP） |
| 512/tcp | exec | 用于对远程执行的进程进行验证 |
| 512/udp | biff [comsat] | 异步邮件客户（biff）和服务（comsat） |
| 513/tcp | login | 远程登录（rlogin） |
| 513/udp | who [whod] | 登录的用户列表 |
| 514/tcp | shell [cmd] | 不必登录的远程 shell（rshell）和远程复制（rcp） |
| 514/udp | syslog | UNIX 系统日志服务 |
| 515 | printer [spooler] | 打印机（lpr）假脱机 |
| 517/udp | talk | 远程对话服务和客户 |
| 518/udp | ntalk | 网络交谈（ntalk），远程对话服务和客户 |
| 519 | utime [unixtime] | UNIX 时间协议（utime） |
| 520/tcp | efs | 扩展文件名服务器（EFS） |
| 520/udp | router [route, routed] | 选路信息协议（RIP） |
| 521 | ripng | 用于互联网协议版本 6（IPv6）的选路信息协议 |
| 525 | timed [timeserver] | 时间守护进程（timed） |
| 526/tcp | tempo [newdate] | Tempo |
| 530/tcp | courier [rpc] | Courier 远程过程调用（RPC）协议 |

续表

| 端 口 号 | 端口服务名称 | 端口说明 |
|---|---|---|
| 531/tcp | conference [chat] | 互联网中继聊天 |
| 532 | netnews | Netnews |
| 533/udp | netwall | 用于紧急广播的 Netwall |
| 535 | iiop | 互联网内部对象请求代理协议（IIOP） |
| 538 | gdomap | GNUstep 分布式对象映射器（GDOMAP） |
| 540/tcp | uucp [uucpd] | UNIX 到 UNIX 复制服务 |
| 543/tcp | klogin | Kerberos 版本 5（v5）远程登录 |
| 544/tcp | kshell | Kerberos 版本 5（v5）远程 shell |
| 546 | dhcpv6-client | 动态主机配置协议（DHCP）版本 6 客户 |
| 547 | dhcpv6-server | 动态主机配置协议（DHCP）版本 6 服务 |
| 548 | afpovertcp | 通过传输控制协议（TCP）的 Appletalk 文件编制协议（AFP） |
| 556 | remotefs [rfs_server, rfs] | Brunhoff 的远程文件系统（RFS） |
| 554 | rtsp | 实时流播协议（RTSP） |
| 563 | nntps | 通过安全套接字层的网络新闻传输协议（NNTPS） |
| 565 | whoami | whoami |
| 587 | submission | 邮件消息提交代理（MSA） |
| 610 | npmp-local | 网络外设管理协议（NPMP）本地/分布式排队系统（DQS） |
| 611 | npmp-gui | 网络外设管理协议（NPMP）GUI/分布式排队系统（DQS） |
| 612 | hmmp-ind | HMMP 指示/DQS |
| 631 | ipp | 互联网打印协议（IPP） |
| 636 | ldaps | 通过安全套接字层的轻型目录访问协议（LDAPS） |
| 674 | acap | 应用程序配置存取协议（ACAP） |
| 694 | ha-cluster | 用于带有高可用性的群集的心跳服务 |
| 749 | kerberos-adm | Kerberos 版本 5（v5）的“kadmin”数据库管理 |
| 750 | kerberos-iv | Kerberos 版本 4（v4）服务 |
| 765 | webster | 网络词典 |
| 767 | phonebook | 网络电话簿 |
| 873 | rsync | rsync 文件传输服务 |
| 992 | telnets | 通过安全套接字层的 Telnet（TelnetS） |
| 993 | imaps | 通过安全套接字层的互联网消息存取协议（IMAPS） |
| 994 | ircs | 通过安全套接字层的互联网中继聊天（IRCS） |
| 995 | pop3s | 通过安全套接字层的邮局协议版本 3（POPS3） |

续表

| 端 口 号 | 端口服务名称 | 端口说明 |
|---|---|---|
| 1080 | socks | SOCKS 网络应用程序代理服务 |
| 1236 | bvcontrol [rmtcfg] | Garcilis Packeten 远程配置服务器[a] |
| 1300 | h323hostcallsc | H.323 电话会议主机电话安全 |
| 1433 | ms-sql-s | Microsoft SQL 服务器 |
| 1434 | ms-sql-m | Microsoft SQL 监视器 |
| 1494 | ica | Citrix ICA 客户 |
| 1512 | wins | Microsoft Windows 互联网名称服务器 |
| 1524 | ingreslock | Ingres 数据库管理系统（DBMS）锁定服务 |
| 1525 | prospero-np | 无特权的 Prospero |
| 1645 | datametrics [old-radius] | Datametrics /从前的 radius 项目 |
| 1646 | sa-msg-port [oldradacct] | sa-msg-port /从前的 radacct 项目 |
| 1649 | kermit | Kermit 文件传输和管理服务 |
| 1701 | l2tp [l2f] | 第 2 层隧道服务（LT2P）/第 2 层转发（L2F） |
| 1718 | h323gatedisc | H.323 电信守门装置发现机制 |
| 1719 | h323gatestat | H.323 电信守门装置状态 |
| 1720 | h323hostcall | H.323 电信主持电话设置 |
| 1758 | tftp-mcast | 小文件 FTP 组播 |
| 1759 | mtftp | 组播小文件 FTP（MTFTP） |
| 1789 | hello | Hello 路由器通信端口 |
| 1812 | radius | Radius 拨号验证和记账服务 |
| 1813 | radius-acct | Radius 记帐 |
| 1911 | mtp | Starlight 网络多媒体传输协议（MTP） |
| 1985 | hsrp | Cisco 热备用路由器协议 |
| 1986 | licensedaemon | Cisco 许可管理守护进程 |
| 1997 | gdp-port | Cisco 网关发现协议（GDP） |
| 2049 | nfs [nfsd] | 网络文件系统（NFS） |
| 2102 | zephyr-srv | Zephyr 通知传输和发送服务器 |
| 2103 | zephyr-clt | Zephyr serv-hm 连接 |
| 2104 | zephyr-hm | Zephyr 主机管理器 |
| 2401 | cvspserver | 并行版本系统（CVS）客户/服务器操作 |
| 2430/tcp | venus | 用于 Coda 文件系统（codacon 端口）的 Venus 缓存管理器 |
| 2430/udp | venus | 用于 Coda 文件系统（callback/wbc interface 界面）的 Venus 缓存管理器 |
| 2431/tcp | venus-se | Venus 传输控制协议（TCP）的副作用 |
| 2431/udp | venus-se | Venus 用户数据报协议（UDP）的副作用 |

续表

| 端 口 号 | 端口服务名称 | 端口说明 |
|---|---|---|
| 2432/udp | codasrv | Coda 文件系统服务器端口 |
| 2433/tcp | codasrv-se | Coda 文件系统 TCP 副作用 |
| 2433/udp | codasrv-se | Coda 文件系统 UDP SFTP 副作用 |
| 2600 | hpstgmgr [zebrasrv] | HPSTGMGR；Zebra 选路[b] |
| 2601 | discp-client [zebra] | discp 客户；Zebra 集成的 shell |
| 2602 | discp-server [ripd] | discp 服务器；选路信息协议守护进程（ripd） |
| 2603 | servicemeter [ripngd] | 服务计量；用于 IPv6 的 RIP 守护进程 |
| 2604 | nsc-ccs [ospfd] | NSC CCS；开放式短路径优先守护进程（ospfd） |
| 2605 | nsc-posa | NSC POSA；边界网络协议守护进程（bgpd） |
| 2606 | netmon [ospf6d] | Dell Netmon；用于 IPv6 的 OSPF 守护进程（ospf6d） |
| 2809 | corbaloc | 公共对象请求代理体系（CORBA）命名服务定位器 |
| 3128/tcp | squid | Squid 万维网代理缓存 |
| 3130 | icpv2 | 互联网缓存协议版本 2（v2）；被 Squid 代理缓存服务器使用 |
| 3306 | mysql | MySQL 数据库服务 |
| 3346 | trnsprntproxy | Trnsprnt 代理 |
| 4011 | pxe | 执行前环境（PXE）服务 |
| 4321 | rwhois | 远程 Whois（rwhois）服务 |
| 4444 | krb524 | Kerberos 版本 5（v5）到版本 4（v4）门票转换器 |
| 5002 | rfe | 无射频以太网（RFE）音频广播系统 |
| 5308 | cfengine | 配置引擎（Cfengine） |
| 5999 | cvsup [CVSup] | CVSup 文件传输和更新工具 |
| 6000 | x11 [X] | X 窗口系统服务 |
| 7000 | afs17-fileserver | Andrew 文件系统（AFS）文件服务器 |
| 7001 | afs17-callback | 用于给缓存管理器回电的 AFS 端口 |
| 7002 | afs17-prserver | AFS 用户和组群数据库 |
| 7003 | afs17-vlserver | AFS 文件卷位置数据库 |
| 7004 | afs17-kaserver | AFS Kerberos 验证服务 |
| 7005 | afs17-volser | AFS 文件卷管理服务器 |
| 7006 | afs17-errors | AFS 错误解释服务 |
| 7007 | afs17-bos | AFS 基本检查进程 |
| 7008 | afs17-update | AFS 服务器到服务器更新器 |
| 7009 | afs17-rmtsys | AFS 远程缓存管理器服务 |
| 9876 | sd | 会话指引器 |

续表

| 端 口 号 | 端口服务名称 | 端口说明 |
|---|---|---|
| 10080 | amanda | 高级 Maryland 自动网络磁盘归档器（Amanda）备份服务 |
| 11371 | pgpkeyserver | 良好隐私（PGP）/GNU 隐私卫士（GPG）公钥服务器 |
| 11720 | h323callsigalt | H.323 调用信号交替 |
| 13720 | bprd | Veritas NetBackup 请求守护进程（bprd） |
| 13721 | bpdbm | Veritas NetBackup 数据库管理器（bpdbm） |
| 13722 | bpjava-msvc | Veritas NetBackup Java/Microsoft Visual C++（MSVC）协议 |
| 13724 | vnetd | Veritas 网络工具 |
| 13782 | bpcd | Vertias NetBackup |
| 13783 | vopied | Veritas VOPIED 协议 |
| 22273 | wnn6 [wnn4] | 假名/汉字转换系统[c] |
| 26000 | quake | Quake（以及相关的）多人游戏服务器 |
| 33434 | traceroute | Traceroute 网络跟踪工具 |

## 17.3 漏洞扫描

Nmap 的确不错，能够轻松地实现 TCP 扫描、UDP 扫描、操作系统指纹扫描、隐蔽扫描等方式，也能得到活动的主机及其端口，还能通过表 17-1 确认该端口对应的网络或者系统服务，从而通过其他方式来获取这些服务的漏洞发起攻击或者是进行网络安全管理。那么，有没有一种比以 Nmap 为代表的端口扫描技术更加具有针对性的漏洞发现技术呢？

### 17.3.1 漏洞扫描基本原理

漏洞扫描就是对计算机系统或者其他网络设备进行安全相关的检测，以找出安全隐患和可被黑客利用的漏洞。显然，漏洞扫描软件是把双刃剑，黑客利用它入侵系统，而系统管理员掌握它以后又可以有效地防范黑客入侵。因此，漏洞扫描是保证系统和网络安全必不可少的手段，必须仔细研究和利用。

漏洞扫描通常采用两种策略，第一种是被动式策略，第二种是主动式策略。所谓被动式策略就是基于主机之上，对系统中不合适的设置、脆弱的口令以及其他同安全规则抵触的对象进行检查；而主动式策略是基于网络的，它通过执行一些脚本文件模拟对系统进行攻击的行为并记录系统的反应，从而发现其中的漏洞。利用被动式策略扫描称为系统安全扫描，利用主动式策略扫描称为网络安全扫描。

下面介绍漏洞扫描的四种检测技术。

- 基于应用的检测技术。它采用被动的、非破坏性的办法检查应用软件包的设置，发现安全漏洞。
- 基于主机的检测技术。它采用被动的、非破坏性的办法对系统进行检测。通常它涉及系统的内核、文件的属性、操作系统的补丁等问题。这种技术还包括口令解密，把一些简单的口令剔除。因此这种技术可以非常准确地定位系统的问题，发现系统的漏洞。其缺点是与平台相关，升级复杂。
- 基于目标的漏洞检测技术。它采用被动的、非破坏性的办法检查系统属性和文件属性，如数据库、注册号等。通过消息文摘算法，对文件的加密数进行检验。这种技术的实现是运行在一个闭环上，不断地处理文件、系统目标。系统目标属性，然后产生检验数，把这些检验数同原来的检验数相比较。一旦发现改变就通知管理员。
- 基于网络的检测技术。它采用积极的、非破坏性的办法来检验系统是否有可能被攻击崩溃。它利用一系列的脚本模拟对系统进行攻击的行为，然后对结果进行分析。它还针对已知的网络漏洞进行检验。网络检测技术常被用来进行穿透实验和安全审记。这种技术可以发现一系列平台的漏洞，也容易安装。但是它可能会影响网络的性能。

优秀的安全扫描产品应该是综合了以上 4 种方法的优点，最大限度地增强漏洞识别的精度。

### 17.3.2　选择：网络漏洞扫描与主机漏洞扫描

基于网络的漏洞扫描器有很多优点。

- 价格方面。基于网络的漏洞扫描器的价格相对来说比较便宜。
- 基于网络的漏洞扫描器在操作过程中，不需要涉及目标系统的管理员。基于网络的漏洞扫描器，在检测过程中，不需要在目标系统上安装任何东西。
- 维护简便。当企业的网络发生了变化的时候，只要某个节点，就能够扫描网络中的全部目标系统，而基于网络的漏洞扫描器不需要进行调整。

相对于前者来说，基于主机的漏洞扫描有以下优点。

- 扫描的漏洞数量多。由于通常在目标系统上安装了一个代理（Agent）或者是服务（Services），以便能够访问所有的文件与进程，这也使得基于主机的漏洞扫描器能够扫描更多的漏洞。这一点在前面已经提到过。
- 集中化管理。基于主机的漏洞扫描器通常都有一个集中的服务器作为扫描服务器。所有扫描的指令均从服务器进行控制，这一点与基于网络的扫描器类似。服务器从下载到最新的代理程序后，再分发给各个代理。这种集中化管理模式，使得基于主机的漏洞扫描器在部署上，能够快速实现。
- 网络流量负载小。由于漏洞扫描器管理器与漏洞扫描器代理之间只有通信的数据包，漏洞扫描部分都由漏洞扫描器代理单独完成，这样大大减少了网络的流量负载。当扫描结束后，漏洞扫描器代理再次与漏洞扫描器管理器进行通信，将扫描结果传送给漏洞扫描器管理器。
- 通信过程中的加密机制。所有的通信过程中的数据包，都经过加密。由于漏洞扫描都在本地完成，漏洞扫描器代理和漏洞扫描器管理器之间，只需要在扫描之前和扫描结束之后建立必要的通信链路。因此，对于配置了防火墙的网络，只需要在防火墙上开

放漏洞扫描器所需的 5600 和 5601 这两个端口，漏洞扫描器即可完成漏洞扫描的工作。然而，基于主机的漏洞扫描工具也存在以下不足之处。

- 价格方面。基于主机的漏洞扫描工具的价格，通常由一个管理器的许可证价格加上目标系统的数量来决定，当一个企业网络中的目标主机较多时，扫描工具的价格就非常高。通常，只有实力强大的公司和政府部门才有能力购买这种漏洞扫描工具。
- 基于主机的漏洞扫描工具，需要在目标主机上安装一个代理或服务，而从管理员的角度来说，并不希望在重要的机器上安装自己不确定的软件。
- 随着所要扫描的网络范围的扩大，在部署基于主机的漏洞扫描工具的代理软件的时候，需要与每个目标系统的用户打交道，必然延长了首次部署的工作周期。

### 17.3.3 高效使用网络漏洞扫描

#### 1．网络漏洞扫描器的构成

网络漏洞扫描器的一种扫描原理和工作原理为：通过远程检测目标主机 TCP/IP 不同端口的服务，记录目标给予的回答。通过这种方法，可以搜集到很多目标主机的各种信息，例如：是否能用匿名登录、是否有可写的 FTP 目录、是否能用 Telnet、httpd 是否用 root 在运行等。在获得目标主机 TCP/IP 端口和其对应的网络访问服务的相关信息后，与网络漏洞扫描系统提供的漏洞库进行匹配，如果满足匹配条件，则视为漏洞存在。此外，通过模拟黑客的进攻手法，对目标主机系统进行攻击性的安全漏洞扫描，如测试弱势口令等，也是扫描模块的实现方法之一。如果模拟攻击成功，则视为漏洞存在。在匹配原理上，该网络漏洞扫描器采用的是基于规则的匹配技术，即根据安全专家对网络系统安全漏洞、黑客攻击案例的分析和系统管理员关于网络系统安全配置的实际经验，形成一套标准的系统漏洞库，然后再在此基础之上构成相应的匹配规则，由程序自动进行系统漏洞扫描的分析工作。所谓基于规则是基于一套由专家根据事先定义的规则的匹配系统，例如，在对 TCP 80 端口的扫描中，如果/cgi-bin/phf/cgi-bin/Count.cgi，根据专家经验以及 CGI 程序的共享性和标准化，可以推知该 WWW 服务存在两个 CGI 漏洞。同时应当说明的是，基于规则的匹配系统也有其局限性，因为作为这类系统的基础的推理规则一般都是根据已知的安全漏洞进行安排和策划的，而对网络系统的很多危险的威胁是来自未知的安全漏洞，这一点和 PC 杀毒很相似。

基于网络的漏洞扫描器通常由以下几个方面组成。

- 漏洞数据库模块。漏洞数据库包含了各种操作系统的各种漏洞信息，以及如何检测漏洞的指令。由于新的漏洞会不断出现，该数据库需要经常更新，以便能够检测到新发现的漏洞。
- 用户配置控制台模块。用户配置控制台与安全管理员进行交互，用来设置要扫描的目标系统，以及扫描哪些漏洞。
- 扫描引擎模块。扫描引擎是扫描器的主要部件。根据用户配置控制台部分的相关设置，扫描引擎组装好相应的数据包，发送到目标系统，将接收到的目标系统的应答数据包与漏洞数据库中的漏洞特征进行比较，来判断所选择的漏洞是否存在。
- 当前活动的扫描知识库模块。通过查看内存中的配置信息，该模块监控当前活动的扫描，将要扫描的漏洞的相关信息提供给扫描引擎，同时还接收扫描引擎返回的扫描结果。

- 结果存储器和报告生成工具。报告生成工具利用当前活动扫描知识库中存储的扫描结果，生成扫描报告。扫描报告将告诉用户配置控制台设置了哪些选项，根据这些设置，扫描结束后，便可知在哪些目标系统上发现了哪些漏洞。

2．基于 B/S 架构的网络漏洞扫描器

基于浏览器/服务器（B/S）的网络漏洞扫描器的结构如图 17-1 所示。这种网络扫描器的工作原理是：当用户通过控制平台发出了扫描命令之后，控制平台即向扫描模块发出相应的扫描请求，扫描模块在接到请求之后立即启动相应的子功能模块，对被扫描主机进行扫描。通过对从被扫描主机返回的信息进行分析判断，扫描模块将扫描结果返回给控制平台，再由控制平台最终呈现给用户。

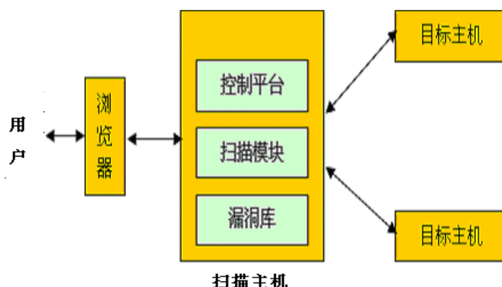



图 17-1　基于 B/S 架构的网络漏洞扫描器结构

3．基于 C/S 架构的网络漏洞扫描器

如图 17-2 所示是采用插件程序结构，客户/服务器模式的网络漏洞扫描器。插件是由脚本语言编写的子程序，扫描程序可以通过调用它来执行漏洞扫描，检测出系统中存在的一个或多个漏洞。添加新的插件就可以使漏洞扫描软件增加新的功能，扫描出更多的漏洞。插件编写规范化后，甚至用户自己都可以用 Perl、C 或自行设计的脚本语言编写的插件来扩充漏洞扫描软件的功能。这种技术使漏洞扫描软件的升级维护变得相对简单，而专用脚本语言的使用也简化了编写新插件的编程工作，使漏洞扫描软件具有强的扩展性。可以针对某一具体漏洞，编写对应的外部测试脚本。通过调用服务检测插件，检测目标主机 TCP/IP 不同端口的服务，并将结果保存在一信息库中，然后调用相应的插件程序，向远程主机发送构造好的数据，并将检测结果也保存于信息库，以提供给其他的脚本运行所需的必要信息，这样可提高检测效率。例如，在针对某 FTP 服务的攻击中，可以首先查看服务检测插件的返回结果，只有在确认目标主机服务器开启 FTP 服务时，对应的针对某 FTP 服务的攻击脚本才能被执行。采用这种插件结构的扫描器，可以让任何人构造自己的攻击测试脚本，而不用去了解太多扫描器的原理。这种扫描器也可以用作模拟黑客攻击的平台。采用这种结构的扫描器具有很强的生命力，如著名的 Nessus 就是采用这种结构，如图 17-2 所示，其中客户端主要设置服务器端的扫描参数，及收集扫描信息。具体扫描工作由服务器来完成。

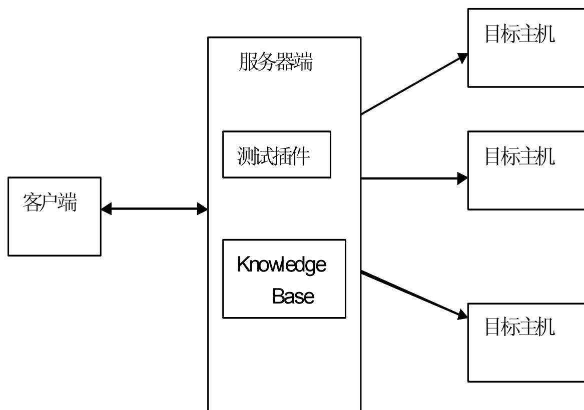



图 17-2　基于 C/S 架构的网络漏洞扫描器结构

### 17.3.4 快速安装 Nessus

#### 1．Nessus 简介

Nessus 是一个功能强大而又易于使用的远程安全扫描器，它不仅免费而且更新极快。安全扫描器的功能是对指定网络进行安全检查，找出该网络是否存在导致对手攻击的安全漏洞。该系统被设计为 Client/Sever 模式，服务器端负责进行安全检查，客户端用来配置管理服务器端。在服务端还采用了 plug-in 的体系，允许用户加入执行特定功能的插件，这些插件可以进行更快速和更复杂的安全检查。在 Nessus 中还采用了一个共享的信息接口，称之知识库，其中保存了前面进行检查的结果。检查的结果可以 HTML、纯文本、LaTeX（一种文本文件格式）等几种格式保存。

在未来的新版本中，Nessus 将会支持更快速的安全检查，而且这种检查将会占用更少的带宽，其中可能会用到集群的技术以提高系统的运行效率。

Nessus 的优点如下。

- Nessus 采用了基于多种安全漏洞的扫描，避免了扫描不完整的情况。
- Nessus 是免费的，比起商业的安全扫描工具如 ISS 具有价格优势。
- Nessus 扩展性强、容易使用、功能强大，可以扫描出多种安全漏洞。

Nessus 的安全检查完全是由 plug-ins 的插件完成的。在 Nessus 主页中不但详细介绍了各种插件的功能，还提供了解决问题的相关方案。有关 plug-in 的详细说明，请查看 http://cgi.nessus.org/plugins/dump.php3?viewby=family。

除了这些插件外，Nessus 还为用户提供了描述攻击类型的脚本语言，来进行附加的安全测试，这种语言称为 Nessus 攻击脚本语言（NSSL），用它来完成插件的编写。

在客户端，用户可以指定运行 Nessus 服务的机器、使用的端口扫描器及测试的内容和测试的 IP 地址范围。Nessus 本身是工作在多线程基础上的，所以用户还可以设置系统同时工作的线程数。这样用户在远端就可以设置 Nessus 的工作配置了。安全检测完成后，服务端将检测结果返回到客户端，客户端生成直观的报告。在这个过程当中，由于服务器向客户端传送的内容是系统的安全弱点，为了防止通信内容受到监听，其传输过程还可以选择加密。

#### 2．安装和启动 Nessus 服务器端

用户可以到 http://www.nessus.org/download.html 下载 Nessus 服务器的最新版本。目前其最新版本为：Nessus-4.2.0-es5.i386.rpmNessus，使用以下命令对其进行安装即可：

```
[root@localhost tmp]# rpm -ivh Nessus-4.2.0-es5.i386.rpm
Preparing...            ########################################### [100%]
1:Nessus                ########################################### [100%]
nessusd (Nessus) 4.2.0 [build K9080] for Linux
(C) 1998 - 2009 Tenable Network Security, Inc.


 - Please run /opt/nessus//sbin/nessus-adduser to add a user
 - Register your Nessus scanner at http://www.nessus.org/register/ to obtain
   all the newest plugins
```

```
  - You can start nessusd by typing /sbin/service nessusd start
```

安装成功后，还需要添加用户来对其进行操作，步骤如下：

```
[root@localhost tmp]# /opt/nessus//sbin/nessus-adduser
//添加用户
Login : root
//设置密码
Login password :
Login password (again) :
Do you want this user to be a Nessus 'admin' user ? (can upload plugins, etc…)
(y/n) [n]: y
User rules
----------
nessusd has a rules system which allows you to restrict the hosts
that root has the right to test. For instance, you may want
him to be able to scan his own host only.

Please see the nessus-adduser manual for the rules syntax

Enter the rules for this user, and enter a BLANK LINE once you are done :
(the user can have an empty rules set)

Login            : root
Password         : ***********
This user will have 'admin' privileges within the Nessus server
Rules            :
Is that ok ? (y/n) [y] y
//添加成功
User added
```

启动 Nessus 非常简单，使用如下命令即可：

```
#/sbin/service nessusd start
```

### 3．安装 Nessus 客户端

Nessus 的客户端有两个版本：Java 版本及 C 版本，Java 版本可以在多个平台中运行，C 版本的支持 Windows。有了这两个客户端的版本就可以在局域网的任何一台机器上进行安全检查了。为了使用的简单性，选择了一款 Windows 系统下的 Nessus 4 客户端版本进行安装和使用，也就是使用 Windows 客户端来控制运行于 Linux 下的 Nessus 服务器端来对局域网中的机器进行漏洞扫描，这也是目前 Nessus 非常流行的一种使用方式。具体的安装如同 Windows 下任何一款

应用软件的安装方式，非常简单，这里不再赘述。

### 17.3.5 使用 Nessus 扫描

下面来看看使用 Nessus 进行扫描的步骤以及效果。通常使用 Nessus 进行扫描需要以下几个步骤。

（1）设置服务器连接。首先需要设置 Nessus 客户端来连接 Nessus 服务器，也就是在上一小节中安装的服务器。在图 17-3 中，配置好相应的主机名和端口，以及登录所需要使用的用户名和密码。

（2）设置 IP 范围。如图 17-4 所示，设置为 IP Range。当然，这里还有其他的选项可提供选择，包括 Single Host、Subnet 等，可以根据实际情况来选择。

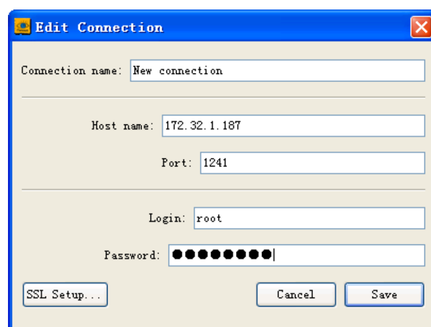



图 17-3 设置服务器连接

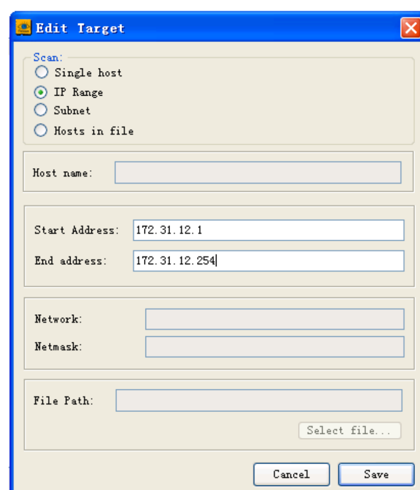



图 17-4 设置扫描的 IP 范围

（3）单击 scan now 按钮，开始对设定范围进行扫描，如图 17-5 所示。

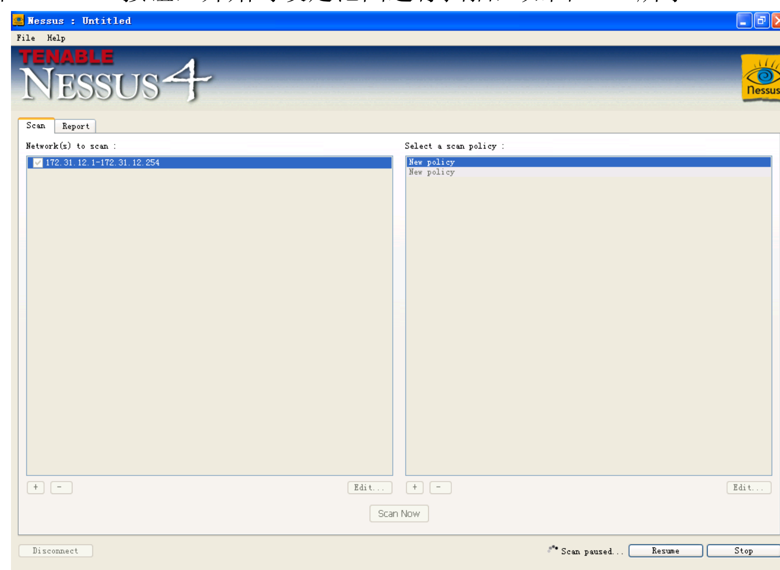



图 17-5 开始扫描

（4）扫描的整体效果。如图 17-6 所示，扫描给出了对 172.31.12.188 这台主机（Linux 操作系统，RHEL 5.0 版本）的扫描结果，可以很清晰地看出操作系统的版本以及开放的端口，同时，也能够将开放的端口的详细信息列出来。

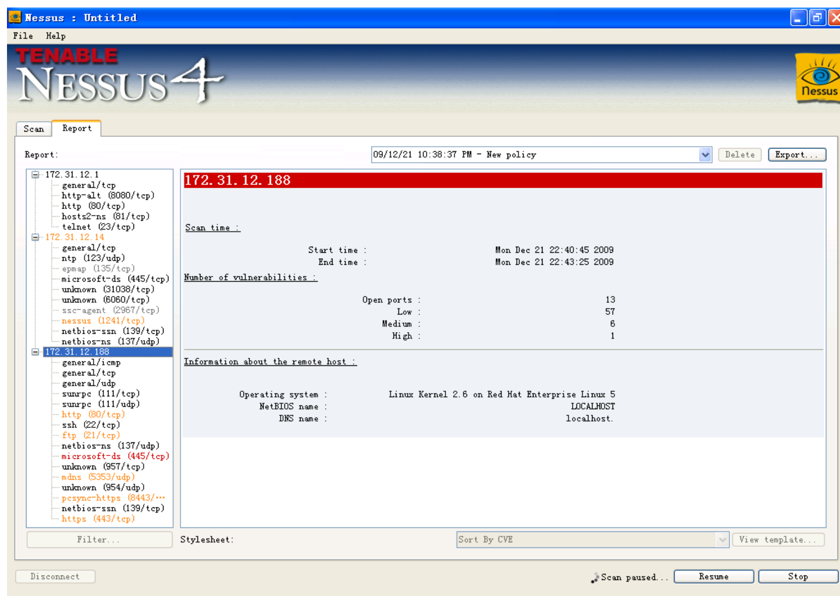



图 17-6 扫描的整体结果

（5）查看具体的漏洞信息。如果想查看具体的漏洞信息报告以及漏洞等级等详细信息时，可以单击图 17-7 所示中的对应开放端口信息，并针对具体信息采取相应的措施来对该漏洞进行修补等操作。

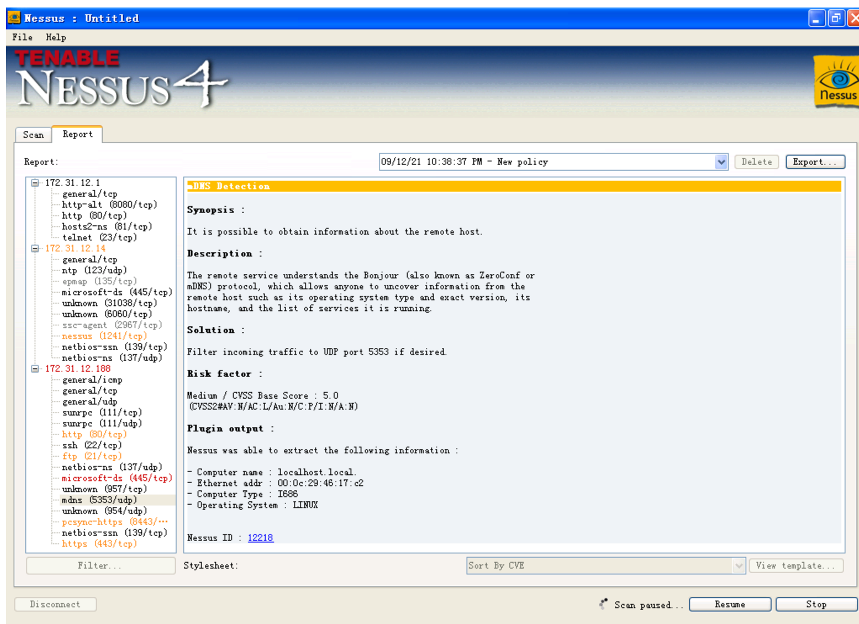



图 17-7 具体的漏洞信息

# 第五篇

# 企业 Linux 安全运维命令、工具

## 工欲善其事，必先利其器

“工欲善其事，必先利其器。”这句名言出自《论语》。孔子告诉子贡，一个做手工或工艺的人，要想把工作完成，并做得完善，应该先把工具准备好。

在企业 Linux 安全运维工作中，其实有很多捷径。这包括掌握一些核心的理念、框架和方法，以及充分地利用现有的开源工具、命令，来灵活、方便、高效地进行运维工作，这样就能提高工作效率。

# 第 18 章

# 终极挑战：企业 Linux 内核构建

## 本章导读

500 强企业在企业级 Linux 应用过程中的一个非常典型和重要的操作就是进行 Linux 内核重构，也就是根据实际的应用需求进行内核定制、裁剪等工作，这是一个非常具有挑战性和专业性的工作，也可以称为一种“终极挑战”，当前众多的大型企业都有自己的 Linux 平台工程师和专业人员来进行该项工作。

基于此原因，本章将对企业级 Linux 内核配置、编译、安装等细节性的问题进行详细介绍。

## 18.1 企业级 Linux 内核简介

安装企业级 Linux 之后，需要重新配置并建立新的 Linux 内核。企业级 Linux 自带了一个预置的内核，简化了安装过程。然而这个内核可能无法正确配置所有的系统功能。因此可能需要通过配置和构建新内核来创建一个针对系统及其特定需求的定制内核。定制内核通常比一般内核要小。但是，编译和维护定制的内核需要大量的时间和工作。

有时候，并不需要构建新内核，只是需要动态改变已安装内核的许多方面。有两种方式可以完成相关改变：一是可使用启动命令加上特定参数，二是修改/etc/sysctl.conf 文件。当系统引导时，sysctl 命令会读取此文件。

也可以将引导命令行使用的相同参数添加到/boot/grub/grub.conf 或其符号链接/etc/grub.conf 中。例如，acpi=off 可禁止启动 acpid（高级配置和电源接口守护进程）。

系统运行时，sysctl 工具可以修改内核参数。此实用工具利用了/proc/sys 设施，它用于定义 sysctl 可以修改的参数。

执行 sysctl -a 命令会显示一个完整的 sysctl 参数清单。下面给出了显示和更改 domainname 内核参数的例子。这个例子中不需要双引号，但必须引用任何字符，否则会被 shell 解释掉。当改变参数时，必须具有 root 权限。

```
#sysctlkernel.domainname
kernel.domainname = tcorp.com

$ su -c 'sysctl -w kernel.domainname="example.com" '
kernel.domainname = example.com
$ sysctlkernel.domainname
kernel.domainname = example.com
```

当构建新内核时，手边需要第一张安装光盘或 DVD。

当构建新 Linux 内核来安装新版本或更改现有版本的配置时，确保手边有第一张安装光盘或 DVD。这样即使完全破坏了系统软件，上述磁盘也可以让你重新启动系统。有无可用的 CD/DVD 意味着慌乱一阵和完全崩溃之间的差异。

在可以开始构建新内核之前，必须下载、安装和预备源代码。此外，还需要建立一个配置文件，描述想要建立的新内核。本章将介绍完成这些任务所需要的步骤。

## 18.2 下载、安装和预备内核源代码

本节介绍如何下载内核源代码到本地系统上。如果要下载企业级 Linux 尚未定制（补丁）的代码，请访问 kernel.org /pub。它还包括预备和安装内核源代码。在完成下载、安装、预备内核源代码之后，需要配置和编译 Linux 内核。

### 18.2.1 先决条件

安装如下软件包：

- rpmdevtools
- yum-utils（随默认系统安装）
- ncurses-devel（使用 make menuconfig 配置内核）
- libglade2-devel（使用 make gconfig 配置内核）

值得注意的是：编译内核需要大量磁盘空间。在编译内核之前，确保有足够的磁盘空间。编译一个默认内核大约需要 3.5GB 空间。在要编译内核的文件系统上，必须有足够的磁盘空间。

### 18.2.2 下载源代码

在下载所需的软件包之后，会运行 rpmdev-setuptree。此实用程序设置了一个 SRPM 构建环境，包括主目录下的 rpmbuild 目录层次结构：

```
$ rpmdev-setuptree
$ lsrpmbuild
BUILD RPMS SOURCES SPECS SRPMS
```

编译内核后，rpmdev-setuptree 创建的 5 个目录包含如下文件：BUILD 中拥有编译内核的结果，RPMS 中拥有编译的 RPM 文件，SOURCES 中拥有源代码和补丁文件，SPECS 中拥有内核的规范文件，SRPMS 中如果有的话，则是源 RPM 文件。

除了安装软件包和安装内核（最后一步）外，不需要也不应该使用 root 权限来配置或构建内核。内核的 README 文件指出：“不要过度使用 root。”

当必须使用 root 权限时，可使用 su–c 构建、定制和安装内核部分。当并不想使用 root 权限时，这种做法也有助于防止使用 root 权限。它还提供了一个更好的记录，用于审查 shell 历史中出现的问题。

要为 RHEL 内核下载源代码，可在浏览器或 FTP 客户端中输入 ftp://ftp.redhat.com/pub/redhat/linux/enterprise，并选择想要的 RHEL 版本，通过链接直接进入 SRPMS 目录。对于 RHEL 6 的服务器内核，可进入：

ftp://ftp.redhat.com/pub/redhat/linux/enterprise/6Server/en/os/SRPMS

从 SRPMS 页面搜索或向下滚动，直到找到 kernel*src.rpm 文件。这时可以单击和下载内核源代码的 RPM 文件。通常情况下，要下载最新版本。这将有一个与 kernel-2.6.32-71.18.1.el6.src.rpm 类似名字的内核。src 表明软件包包含的源文件。另外，如果有 Red Hat 网络账户，可以从 rhn.redhat.com 下载内核源代码 RPM 文件。

本节例子中所示的 Sam，其主目录是/home/sam（~sam），使用 kernel-2.6.35.11-83.fc14 内核。构建内核时，你可能工作在主目录中，使用的内核名称也可能不同于 Sam 的内核名称。此时只要遵循例子进行必要的替换就可以了。

### 18.2.3 安装源代码

当内核源代码 RPM 包文件位于主目录时，运行 yumbuilddep 可下载和安装需要构建源代码

包（依赖）的软件包。因为使用安装包，所以必须具有 root 权限。在下面的例子可使用用户名和正在使用的内核名称替换 sam 和 2.6.35.11-83.fc14，来指定 RPM 文件在系统中的位置。

```
$ su -c 'yum-builddep ~sam/kernel-2.6.38.2-9.fc18.src.rpm'
password:
Loaded plugins: langpacks, presto, refresh-packagekit
Adding en_US to language list
Getting requirements for kernel-2.6.38.2-9.fc18.src
…
Install 29 Package(s)


Total download size: 26 M
Installed size: 77 M
Is this ok [y/N]: y
...
Complete!
```

最后，以非特权用户角色在主目录中工作时，可使用 RPM 在主目录下的 rpmbuild 目录中安装内核源代码：

```
$ rpm -Uvh kernel-2.6.38.2-9.fc18.src.rpm
1:kernel warning: user mockbuild does not exist - using root
warning: group mockbuild does not exist - using root
warning: user mockbuild does not exist - using root
…
1:kernel ########################################### [100%]
```

用户可以忽略上述错误，它们都是一些关于不存在的组和用户的错误。

### 18.2.4　预备源代码

可以运行程序前的最后一步是预备源代码。预备过程会解开压缩文件和适用的补丁。下面使用 rpmbuild–bp 命令预备源代码：

```
$ cd /home/sam/rpmbuild/SPECS
$ rpmbuild -bp --target $(arch) kernel.spec
Building target platforms: i686
Building for target i686
Executing(%prep): /bin/sh -e /var/tmp/rpm-tmp.SJXxg1
+ umask 022
+ cd /home/sam/rpmbuild/BUILD
…
```

如果 rpmbuild 找不到它需要预备代码的包，会显示如下消息：

```
error: Failed build dependencies:
elfutils-libelf-devel is needed by kernel-2.6.32-71.18.1.el6.i686
zlib-devel is needed by kernel-2.6.32-71.18.1.el6.i686
binutils-devel is needed by kernel-2.6.32-71.18.1.el6.i686
```

在这种情况下，安装所需的软件包（可能需要安装与例子中不同的包）：

```
$ su -c 'yum install elfutils-libelf-develzlib-develbinutils-devel'
```

安装软件包后，再次使用前面的 rpmbuild 命令。在本地系统上安装和预备好内核源代码后，就可以配置、编译并安装它。

## 18.3 源代码配置和编译 Linux 内核

按上节所述安装和预备源代码后，Sam 的源代码在/home/sam/rpmbuild/BUILD/kernel-2.6.38.fc15/linux-2.6.38.i686 目录，其内核规范文件在/home/sam/rpmbuild/SPECS。程序源代码将在具有类似名称的目录（~/rpmbuild/BUILD/kernel*/linux*），而内核规范文件则在~/rpmbuild/SPECS 目录。

本节的大部分工作是在源代码所在的目录（linux*）中做的。在 SPECS 目录中标记和安装内核。

### 18.3.1 标记内核

为了防止覆盖现有的内核文件，并确定内核的各种汇编版本，可以改变 kernel.spec 文件中的 buildid 变量，这个变量的初始值为.local。无论给这个变量赋什么值都需要将其放置在内核名称和发行号的末尾，以便确定内核。可以此字符串中应用到内核的补丁标记来帮助人们追查发生的问题。使用 cd 可把目录切换到~/rpmbuild/SPECS，此时编辑 kernel.spec，并改变行中的 local 即可：

```
#% define buildid .local
```

要标识正在构建的内核值，还必须删除前导哈希标记（#）、哈希标记后面的空格以及%和 define 之间的空格：

```
%define buildid .sam1104181000
```

### 18.3.2 .config：配置内核

在编译代码并创建 Linux 内核之前，必须做出决定，并指定想要的内核支持哪些功能。可以按以下两种方式之一来配置内核支持大多数功能：把该功能构建到内核中或指定该功能作为可加载的内核模块。在决定采用方法之前，必须权衡内核的大小，以及加载模块所花费的时间。要使内核尽可能小，同时又最大限度地减少模块的加载时间。

~/rpmbuild/BUILD/kernel*/linux*目录中的.config 文件控制新内核支持哪些特性，以及如何支持。下一节提供了取代现有自定义内核的指令。“自定义内核”介绍了当.config 文件不存在时如何创建它的默认版本，以及当它存在时如何进行编辑。

如果已经配置了一个定制的内核，可能需要用类似配置（新内核）来替换它。每个内核都可

能有新的配置选项，这也说明为什么使用旧的.config 文件编译新内核是不好的做法。本节将介绍如何升级现有的.config 文件，所以该文件中将包含新内核的新选项，并保持旧选项的现有配置。

上述工作可在~/rpmbuild/BUILD/kernel*/linux*目录下完成。系统为运行在/boot 下的本地系统内核保存了配置文件的副本。下面的命令将会把此文件复制到工作目录中的.config：

```
$ pwd
/home/sam/rpmbuild/BUILD/kernel-2.6.38.fc15/linux-2.6.38.i686
$ cp /boot/config-$(uname -r) .config
```

在此命令中，shell 执行 uname -r 并使用命令替换，以该命令的输出取代$（uname -r），这也是在本地系统上运行内核发行版的名称。

接下来使用命令 make oldconfig 加新内核选项可修补.config 文件，这些选项在旧内核中并不存在。该命令显示了每个内核选项在新、旧内核中是相同的，并以旧内核同样的设置方式自动设置新内核选项的状态。当发现一个选项只出现在新内核中，而不在旧内核中时，该命令就会停止，然后会显示类似[N/y/?]（NEW）的提示，同时显示可能的响应，并表明此选项是新的。提示符显示大写字母的默认响应，可以输入此字母（大写或小写），然后按回车键，或只按回车键来响应。在这个例子中，Tickless 系统选项是新的，且默认响应是 Y，包括新内核的选项。要选择一个非默认响应（n 表示 no，不包括选项，m 意味着包括作为模块的选项），必须输入该字母，然后按回车键。输入“?”然后按回车键来显示有关选项的更多信息。

```
$ make oldconfig
scripts/kconfig/conf -oldconfigKconfig
*
* Linux Kernel Configuration
*
*
* Code maturity level options
*
Prompt for development and/or incomplete code/drivers (EXPERIMENTAL) [Y/n/?] y
*
* General setup
*
Local version - append to kernel release (LOCALVERSION) []
Automatically append version information to version string (LOCALVERSION_AUTO)
[N/y/?] n
…
*
* Processor type and features
*
Tickless System (Dynamic Ticks) (NO_HZ) [Y/n/?] (NEW) ? ?
```

```
This option enables a tickless system: timer interrupts will
only trigger on an as-needed basis both when the system is
busy and when the system is idle.

Tickless System (Dynamic Ticks) (NO_HZ) [Y/n/?] (NEW) ? RETURN
High Resolution Timer Support (HIGH_RES_TIMERS) [Y/n/?] y
Symmetric multi-processing support (SMP) [Y/n/?] y
Subarchitecture Type
> 1. PC-compatible (X86_PC)
2. AMD Elan (X86_ELAN)
…
#
# configuration written to .config
#
```

### 18.3.3　定制内核

~/rpmbuild/BUILD/kernel*/linux*目录和 configs 子目录为各种处理器、多处理器和配置提供示例配置文件。在开始之前你可能想看看这些文件，或使用其中之一作为模板。要使用这些文件之一，将它复制到.config。

可以使用三个标准命令之一，建立配置 Linux 内核的.config 文件：

```
$ make config
$ make menuconfig
$ make gconfig
```

下面需要为所需的软件包列表运行所有这些命令，而不是这些命令中的第一个。

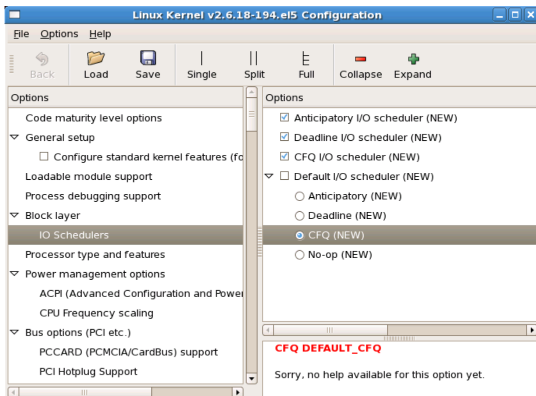
如果.config 文件不在工作目录中，这些命令将首先创建与本地系统运行内核相匹配的.config 文件，然后修改该配置。只有本地运行内核的配置文件位于/boot/config-$（uname -r）中，该命令才可以设置这个.config 文件。如果想使用与现有内核类似的配置，需要建立一个新的内核，相关内容请参阅前一节。

make config 命令是三个命令中最简单的一个，它使用文本接口，且不需要额外的软件。相对来说，使用配置界面是最难的。make menuconfig 命令使用伪图形界面，同时还显示文本界面。make gconfig 命令使用 GTK +（www.gtk.org），并显示图形界面。

每个命令都会提出相同问题，产生相同结果，并给予相同答复。第一个和第二个命令工作在基于字符的环境下，第三个命令则工作在图形环境下。对于使用 GUI 工作的很多管理员来说，第三种方法是最容易使用的。

make gconfig 命令显示了 Linux 内核配置窗口，可以查看三种配置：单一的、分割的或完整的视图。通过单击工具栏中 save（保存）图标右侧的三个图标之一可以选择视图。图 18-1 显示了分割视图。在这种视图中，左框架显示选项，右上角列出了每个选项的功能。右下角的视图描

述突出显示的选项或功能。图 18-2 则显示了完整视图。



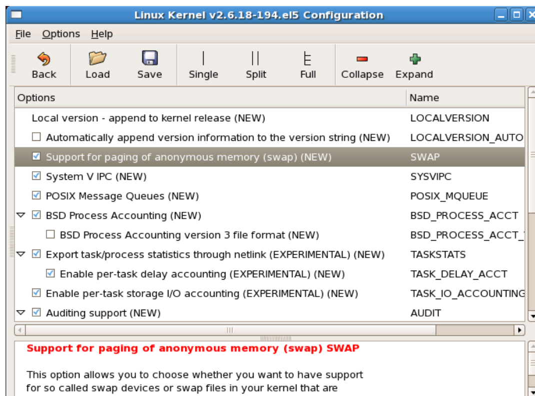




图 18-1　Linux 内核配置窗口，分割视图　　　　图 18-2　Linux 内核配置窗口，完整视图

在所有视图的最右侧是 N、M、Y 和 Value 四列。如果未使用那行的功能，N 出现在 N 列。M 列中的 M 意味着该功能是作为一个模块包含的。Y 列中的 Y 意味着该功能被编译在内核中。下划线在这些列的任何地方，意味着出现在该列的值是有效的功能（例如，下划线在 M 表示该功能可以作为一个模块包含其中，虽然它现在还没有包含进去）。Value 列中包含该功能的值（N、M 或 Y）。

在任何视图中，都可以双击功能旁边的复选框和单选按钮，选择或取消该功能。一个空的复选框、单选按钮表示该功能被禁用，打钩表示它将被编译进内核，减号意味着它被编译成一个模块。突出显示一个选择后，也可以按 M 代表模块，N 代表不包括以及 Y 代表编译进内核。选择 Options→Show All Options 菜单命令，以显示所有选项和功能。

下面介绍检查选项并标记想要配置进新内核的功能。在配置过程的任何时间都可以把当前定义的配置存储到一个文件中，从文件加载配置，保存更改退出或不保存更改退出。操作选项位于菜单栏的 File 中。完成后，选择 File→Save 菜单命令并关闭窗口。

## 18.3.4　清理源代码树

在生成.config 文件之后，编译或重新编译内核之前，可使用以下命令清除所有潜在的陈旧的*.o 文件的源代码树：

```
$ make clean
CLEAN arch/x86/boot/compressed
CLEAN arch/x86/boot
CLEAN /home/sam/rpmbuild/BUILD/kernel-2.6.38.fc15/linux-2.6.38.i686
CLEAN arch/x86/kernel/acpi/realmode
CLEAN arch/x86/kernel/cpu
...
CLEAN Documentation/watchdog/src
CLEAN .tmp_versions
CLEAN vmlinuxSystem.map .tmp_kallsyms3.o .tmp_kallsyms3.S ...
```

当编译内核时，此命令确保正确应用所使用的任何编号方案。

### 18.3.5 复制配置文件

在编译内核之前，必须把要使用的.config 文件复制到用于编译内核的目录，这个文件必须有一个与本地系统上使用内核对应的名称，例如，config-i686-generic：

```
$ cp .config ~/rpmbuild/SOURCES/config-$(arch)-generic
```

### 18.3.6 编译内核映像文件和可加载模块

接下来的命令将编译内核和模块。编译内核可能需要几个小时以上，实际取决于所用系统的性能。

```
    $ pwd
    /home/sam/rpmbuild/SPECS
    $ rpmbuild -bb --target $(arch) kernel.spec
    Building target platforms: i686
    Building for target i686
    Executing(%prep): /bin/sh -e /var/tmp/rpm-tmp.JoAIFx
    + umask 022
    + cd /home/sam/rpmbuild/BUILD
    …
    Wrote:  /home/sam/rpmbuild/RPMS/i686/kernel-debug-devel-2.6.38.2-9.sam1 …
fc18.i686.rpm
    Wrote:  /home/sam/rpmbuild/RPMS/i686/kernel-debug-debuginfo-2.6.38.2-9.s …
fc18.i686.rpm
    Executing(%clean): /bin/sh -e /var/tmp/rpm-tmp.gZcEvK
    + umask 022
    + cd /home/sam/rpmbuild/BUILD
    + cd kernel-2.6.38.fc15
    + rm -rf /home/sam/rpmbuild/BUILDROOT/kernel-2.6.38.2-9.sam1104181000.
fc18.i386
    + exit 0
```

### 18.3.7 使用可加载内核模块

可加载内核模块（有时也被称为模块或可加载模块）是一个对象文件（内核的一部分），它是在运行时被链接到内核的。几乎在任何时间，都可以从正在运行的内核中插入或删除模块（正在使用的模块除外）。这种能力使内核具有灵活性，也使得内核在任何给定的时间都尽可能小。模块是处理一些内核功能的很好的方法，包括非持续使用的驱动程序（如磁带驱动器）。模块文件名以.ko 结束，并存储在/lib/modules 的子目录中。在企业级 Linux 中，内核模块会与前一节中

解释的内核一起被编译。

表 18-1 列出了一些可用的工具来帮助使用模块。相关选项和更多信息请参阅相应的手册页。

表 18-1　使用模块的工具

| 工具/程序 | 功　能 |
| --- | --- |
| depmod | 兼容模块的依赖关系 |
| insmod | 在运行的内核中加载模块 |
| lsmod | 列出所有已加载模块的信息 |
| modinfo | 列出有关模块的信息 |
| modprobe | 关于模块的加载、卸载和报告。当加载一个模块，它也将加载依赖关系 |
| rmmod | 从正在运行的内核中卸载模块 |

## 18.4 安装内核、模块和相关文件

下一步必须以 root 权限执行，需要复制已编译的内核、模块和相关文件到适当的目录，这个目录通常是/boot 和/lib/modules 的一个子目录。当把分区挂载在/boot 时，文件都保存在该分区的根目录（/boot）中。因为已经创建了一个 RPM 包，所以安装这些文件是很容易的。下面的命令将从 Sam 主目录的 rpmbuild 目录层次结构安装新内核文件到正确的目录：

```
$ su -c 'rpm -ivh --force ~sam/rpmbuild/RPMS/i686/kernel-2.6.38.2-9.sam110418
1000.fc18.i686.rpm'Password:
Preparing... ########################################### [100%]
1:kernel ########################################### [100%]
```

以这种方式安装内核可以更新 grub.confGRUB 配置文件以包含新内核。

通过单击屏幕右上角的用户名并选择 Shut Down，然后选择 Restart，就可以重新启动系统。如果在控制台上工作，可按 CONTROL-ALT-DEL。还可以从控制台、基于字符的终端或终端仿真器使用 reboot 命令。当系统重新启动后，可以使用 uname 验证加载的新内核：

```
$ uname -r
2.6.38.2-9.sam1104181000.fc18.i686
```

## 18.5 Linux 系统故障处理

在进行内核编译、裁剪或者是启动 Linux 系统的过程中，可能会遇到形形色色的问题，是频繁地重新启动机器，还是重新安装系统呢？其实，这些都是比较笨的办法，在 Linux 系统中，有很多办法可以挽救崩溃的系统。下面将给出在实际使用中最为典型的四种场景，来一一介绍如何成功挽救崩溃的系统。

### 18.5.1 修复文件系统

虽然在引导过程中会自动检查文件系统，但如果出了问题不能自动修复，将不得不手动检查。默认情况下，fsck 在开机时不能自动修复文件系统，需要让 Linux 进入单用户模式，这样就可以手动运行 fsck。如果有必要，可以显式启动系统到单用户模式。不要挂载除根以外的任何 Linux 自动挂载设备。

值得注意的是：在运行 fsck 之前要备份严重损坏的文件系统（当一个文件系统严重损坏而试图修复它时），fsck 有时会使情况变得更糟。在这种情况下，在试图修复它之前，通过从损坏的文件系统中复制可读数据，就有可能恢复更多的数据。当损坏的文件系统保存了重要数据时，在试图使用 fsck 修复它之前，先使用 dd 对系统做一个完整的二进制备份。

在单用户模式下执行 mount 命令，以确保要检查的本地文件系统没有被挂载。然后在这些文件系统上运行 fsck，根据需要对其进行修复。请注意告知被修复（并能识别）的任何普通文件或目录的所有者，这些文件可能会不完整或不正确。在每个文件系统中为丢失的文件查找 lost+found 目录。成功后运行 fsck，如果系统自动进入单用户模式，输入 exit 从单用户 shell 退出，并恢复引导该系统，否则执行 reboot 命令。

如果文件不正确或完全丢失，则必须从文件系统的备份副本中重建它们。

### 18.5.2 重新安装 MBR

当 Windows 安装覆盖了 MBR 时，有必要重新安装它，从安装 DVD 引导系统，并选择 Rescue installed system。然后挂载系统映像并在适当的设备上运行 grub-install：

```
# chroot /mnt/sysimage
# grub-install /dev/sda
```

### 18.5.3 当系统无法引导时

当系统无法从硬盘引导时，引导系统到挽救已安装系统或到单用户/挽救模式。如果系统启动，在硬盘的根文件系统上运行 fsck，并尝试再次从硬盘引导。如果系统仍然无法启动，可能需要重新安装主引导记录。

当一切都不行时，在安装过程中执行“升级”到 Linux 的目前版本。Fedora/RHEL 系统可以执行一个非破坏性的升级，并可以在此过程中修复部分损坏内容。

### 18.5.4 挽救已安装的系统

当挽救已安装系统时，可以修复一个不能正常引导的系统：更改或替换配置文件、使用 fsck 检查并修复分区、重写引导信息以及更多。为了挽救已安装系统，从网络引导 CD 或安装 DVD 引导系统，并从“欢迎”菜单中选择 Rescue installed system。当系统要求指定要使用的语言和键盘时，随后就会显示挽救画面，如图 18-3 所示。

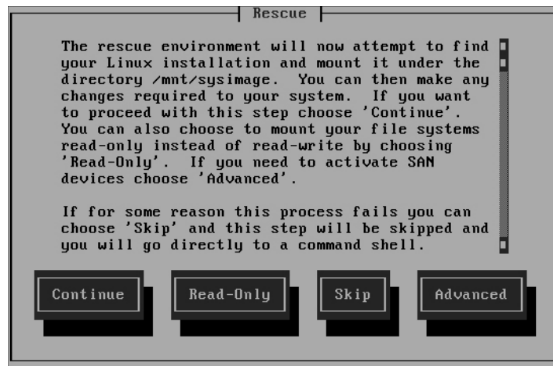



图 18-3　挽救画面

挽救画面首先会询问是否要设置网络接口。如果想从局域网上的其他系统中复制文件或从互联网上下载文件，这个接口是必需的。选择设置该网络接口时，需要决定是否让 DHCP 自动配置网络连接还是手动提供接口的 IP 地址和网络掩码以及网关和 DNS 服务器的 IP 地址。

如果挽救过程发现已有 Linux 安装，可以选择将其挂载到/mnt/sysimage 下，选择只读模式。随着现有安装的挂载，一旦系统显示 shell 提示符（类似于 bash-4.2#），可以执行 chroot/mnt/sysimage 命令以访问现有安装，就如同它已经被正常引导了一样，现有安装的根目录为/（根）。如果选择不挂载现有安装，则使用挂载在标准位置（/bin、/usr/bin 等）的标准工具运行挽救系统。可以从本地安装修复或挂载分区。当从挽救 shell 退出后，系统将重新启动。如果想从硬盘引导，需要取出 CD 或 DVD。

# 第 19 章

# 神兵利器：企业 Linux 数据备份及安全工具

本章导读

除了本书在前面章节介绍所提到的一些安全工具和数据备份工具外，本章还将对一些比较常用的优秀开源工具进行介绍。500 强企业在进行 Linux 安全管理和运维的过程中，可以很好地利用这些开源工具来保证企业信息安全。

# 19.1 安全备份工具

## 19.1.1 Amanda

Amanda（Advanced Maryland Automatic Network Disk Archiver），马里兰高级自动网络磁盘存档工具）是由马里兰大学的 James da Silva 在 1991 年所开发的。它是一个复杂的网络备份系统，能够把 LAN 中的所有计算机备份到一台服务器的磁带驱动器、磁盘或光盘上。Amanda 本身并不是备份程序，它其实只是管理其他备份软件的封装软件。它使用系统上的 dump 和 restore 命令作为底层的备份软件，同时也能够使用 tar 命令，针对于 Windows 计算机，Amanda 还能够使用 smbtar 命令来实现备份。

Amanda 支持类型广泛的磁带驱动器，并且能够使用磁带驱动器中的硬件压缩功能，或者也可以在数据通过网络之前使用客户机的 compress 以及 gzip 命令来压缩备份。其次，Amanda 能够使用临时保存磁盘作为备份存档的中间存储媒介，以优化磁带的写入性能并保证在磁带出错时也能备份数据。

Amanda 是当前最流行的免费备份解决方案。目前 Amanda 最新的稳定版本是 2.6.1，是 2009 年 1 月 26 日发布的。用户可以通过 http://www.amanda.org 站点免费获得。

## 19.1.2 BackupPC

BackupPC 是一个高效的企业级别的备份系统，可以备份桌面系统、笔记本等。可以从网站 http://backuppc.sourceforge.net/访问并下载。它支持 Linux 和 Windows 操作系统的备份，目前常用于 Linux 操作系统。其最新的版本为 3.1.0。它具有以下几个显著特点。

- 最小化使用服务器的磁盘空间和 I/O 来对数据进行备份和恢复。
- 采用优化的压缩技术，进一步减少磁盘空间需求量。
- 具有强大和友好的图形用户界面，方便用户操作使用。
- 支持多种备份方式和备份选项。
- 灵活的配置方法。

## 19.1.3 Bacula

Bacula 采用模块化设计，采用 C/S 构架，理论上可以把任意多台主机的资料备份到任意多台机器中，而用户不需要在每台机器上都写一个配置文件控制它们的运作，所有主要的工作都在一台主控备份机器上进行。登录主控备份，用户就可以清楚地知道和监控哪些备份正在运行、哪些备份成功了、哪些备份失败了，所有的备份日志也会集中到用户指定的地方，这样的集中式管理机制让管理工作更简单了。恢复的时候也很简单，简单运行几个命令，用户就可以把指定的备份恢复。支持完全备份、差异备份、增量备份等多种备份机制；支持把备份写到多种备份媒质中，比如写入到硬盘文件中，也支持写到磁带中。支持平台相当多，设置包括 Win 平台（备份 Win，还不支持备份到 Win）。其网站为：http://www.bacula.org/en/。目前其最稳定版本为 2.4.4（2009 年 1 月发布），而最新版本为 bacula-2.5.42-b2，它是一个 Beta 测试版本，于 2009 年 3 月发布。

## 19.1.4 Xtar

Xtar 是 Linux 系统桌面环境下查看和处理 tar 的工具。tar 工具是 UNIX 备份文件的工具，Linux 继承了该工具。因为 tar 几乎可以工作于任何环境中，所以 Linux 老用户都信赖它。但是 tar 是一个命令行的工具，没有图形界面，使用起来不是很方便。tar 命令的参数非常多，常用的包括 30 多个，初学者往往不易掌握。而 Xtar 是一个图形化的 tar 工具。Xtar 全部兼容 tar 命令，可以在桌面环境下完成打包整个目录树的任务，这使得它特别适合用于备份。Xtar 与文件系统无关，可以使用在 ext2、ext3、JFS、Reiser 和其他文件系统上，支持各种备份介质：软盘、光盘、可重写的光盘、JazZip、磁带等。

## 19.1.5 Taper

Taper 是一款拥有良好用户界面的磁带备份和恢复软件，并且是开放源代码的。它可以从一台磁带机上备份/恢复软件，支持自动更新备份和恢复，是一个相当好用的工具。Taper 可以运行在命令行和 KDE、GONME 桌面环境下。在命令行下启动一个终端输入即可启动。Taper 最大的优点是支持多种备份介质：磁带、软盘、ZIP 驱动器、硬盘等。毕竟磁带机的价格还是比较昂贵的。缺点是 Taper 最新版本一次备份文件最大容量不能超过 4GB。该软件的下载网站为：taper.sourceforge.net/。目前，其最新版本为：6.9b。

## 19.1.6 Arkeia

Arkeia 是一个基于客户/服务模式的备份解决方案。Arkeia 使用了独特的多线程技术，因此备份速度很快。如果客户端是选择用图形界面的，需要 Java 虚拟机（JVM）支持。作为商业软件 Arkeia 和 Kdat 相比最大的优点是备份速度快（在高速网络中 Arkeia 一个小时可以备份 700GB 文件）和广泛应用的平台（通过 Samba 可以备份 Windows 主机）。如果使用图形界面需要安装两个软件：arkeiasb-server、arkeiasb-gui。Arkeia 有非常详细的在线帮助，很容易上手。其下载网站为：www.arkeia.com/download/。

## 19.1.7 webCDcreator

目前政府和企事业单位改换为 Linux 用于桌面系统。针对这种情况，若想实现备份，可以在安装刻录机的 Linux 服务器部署 webCDcreator 共享刻录软件。然后网络中的所有节点（Linux、Windows 均可）都可使用这台刻录机进行数据备份。首先在服务器端安装配置软件，然后在客户端通过安装 Java 的浏览器即可访问服务器上的刻录机。部署 webCDcreator 刻录软件的优点是：节约资金，因为所有软件都是免费的。所使用的机器只需能够运行内核 2.2.x 以上的任一 Linux 发行版。便于管理，集中化的方式有利于管理。其下载网站为：http://joerghaeger.de/webCDcreator/。

## 19.1.8 Ghost for Linux

Ghost for Linux 是一个出色的硬盘对拷工具，它可以最大限度地减少用户每次安装操作系统的时间。它能适用于不同大小的硬盘，而且空间都被充分利用起来。注意 Ghost for Linux 不是赛门铁克的 Ghost，它是一个开放源代码软件，支持 ATA、serial-ATA 和 SCSI 硬盘。使用 Ghost for Linux 进行 Linux 系统备份的优点在于：无须购买磁带机设备，使用一个大硬盘就可以为多台系

统进行全盘备份（一台磁带机的价格相当于 100 个硬盘的价格）。可以使用匿名传输的方式把备份文件传输到 FTP 服务器。缺点在于：备份时间比磁带机长。其下载网站为：https://sourceforge.net/projects/g4l。

### 19.1.9 NeroLINUX

随着 Linux 内核版本的增加，Linux 操作系统越来越大，用 DVD 刻录机备份操作系统成为必然。NeroLINUX 是 Nero 公司发布的一款桌面环境下的 CD/DVD 刻录软件，在操作上 NeroLINUX 与 Windows 的版本基本相同，它支持 2.4/2.6 版本内核，并且将采用 RPM 和 Debian 包等易安装软件包形式。从备份介质方面，NeroLINUX 支持所有主流刻录盘片。它有非常容易使用的界面，功能比较齐全。缺点是目前 NeroLINUX 还只有 RPM 和 Deb 两种版本，没有提供源代码，所以只能用于使用以上两种发行版本的 Linux。其下载网站为：www.nero.com。

### 19.1.10 mkCDrec

mkCDrec（建立可恢复系统的只读 CD）是 Linux 系统中建立可引导灾难恢复 CD 的工具。它支持多数文件系统、RAID（廉价冗余磁盘阵列）以及可选的工具，可以用来进行系统故障修复。使用 mkCDrec CD 来引导系统，用户还需做其他工作：如果用户选择建立系统备份，那么用户可以执行/etc/recovery/start-restore.sh 命令，该命令运行后将提示用户完整的系统恢复过程。如果您在建立启动 CD 时没有进行备份，那么系统会提示没有数据可以进行恢复。/etc/recovery 目录包含了许多工具可以用来进行系统恢复或者修复崩溃的服务器，在 mkCDrec 的 Web 站点上有详细的说明。其下载网站为：http://mkcdrec.ota.be。

## 19.2 Sudo：系统管理工具

Sudo 是一款开源安全工具，常用于对 Root 权限进行控制和审计。它的指导思想是“在保证人们能正常工作的条件下，尽量压缩授予他们的权限”。系统管理员不仅可以让指定的用户或用户组作为 root 用户或其他用户来运行某些命令，还能将指定的用户所输入的命令和参数作详细的记录。当然，该软件是可以免费的，具体地址是 www.gratisoft.us/sudo/download.html。

Sudo 程序是一款在命令行方式下工作的安全工具，并且我们每次只执行一条命令。Sudo 支持的功能如下。

- 命令日志。记录命令和参数。该功能用于跟踪用户输入的命令，尤其适合于进行系统审计。因为 Sudo 会记录下所有作为 root 用户（或者规定的其他用户）的命令，所以许多管理员经常用它来替代 root shell，以便记录下自己使用的命令，这不仅能增进系统安全，还能用来进行故障检修。
- 集中记录多个系统的日志。Sudo 联合系统日志守护进程 syslog 后，能将所有日志集中存放在一个主机上。
- 命令限制。限定用户或者用户组能够使用的命令。
- 检票系统。检票系统通过在用户登录到 Sudo 时所创建的票据来设定时间限制。票据只在规定的时间内有效。每个新的命令都会刷新票据的预设时间，默认的预设时间是 5

分钟。现实中，该功能非常有用，有了它，即使 root 用户离开系统时忘了注销，也不至于被其他可以接触到键盘的用户肆无忌惮地窥探系统。因为票据过期后，系统必须重新登录。所以，我们最好把有效时间尽量设得短一点，如默认有效时间为 5 分钟。检票系统还可以用来清除用户的票据文件。

- 集中管理多个系统。Sudo 的配置一般写在/etc/sudoers 文件中，而该文件可以供多个系统使用，这样一来，我们就可以在一个主机上对这些系统进行集中管理了。

Sudo 几乎支持所有的 UNIX 操作系统版本，但如果要从源代码进行安装的话，必须准备好 C 编译器和 make 工具。

## 19.3 NetCat：网络安全界的瑞士军刀

NetCat是一款非常简单的UNIX工具，可以读、写TCP或UDP网络连接（network connection）。它被设计成一个可靠的后端（back-end）工具，能被其他的程序或脚本直接地或容易地驱动。同时，它又是一款功能丰富的网络调试和开发工具，因为它可以建立你可能用到的几乎任何类型的连接，以及一些非常有意思的内建功能。NetCat 的实际可运行的名字叫 nc，应该很早就被提供，就像另一个没有公开但是标准的 UNIX 工具。

NetCat 可以发现网络上的设备并且绘制网络地图。人们可以从 Sourceforge 网站和@stake 网站下载该工具。目前，@stake 大约提供二十多种安全工具，工具种类涉及信息采集、证据收集、网络实用工具、口令审计、恢复与修补以及安全漏洞扫描等。其中，Andreas Junestam 编写的 WFPdisable、Paul Clip 编写的 AUSTIN、Frederic Bret-Mount 开发的 ComBust 以及 Ollie Whitehouse 开发的 WAP Assessment Tool 都以保持最新状态而闻名

其实NetCat 的原理很简单，它就是从网络的一端读入数据，然后输出到网络的另一端，它可以使用 TCP 和 UDP 协议。之所以叫做 NetCat，因为它是网络上的 cat，想象一下 cat 的功能，读出一个文件的内容，然后输出到屏幕上（默认的 stdout 是屏幕，当然可以重定向到其他地方）。NetCat 也是如此，它读取一端的输入，然后传送到网络的另一端。

NetCat 的一些主要功能如下。

- 支持连出和连入（outbound and inbound connection）、TCP 和 UDP、任意源和目的端口。
- 全部 DNS 正向/反向检查，给出恰当的警告。
- 使用任何源端口。
- 使用任何本地设置的网络资源地址。
- 内建端口扫描功能，带有随机数发生器。
- 内建 loose source-routing 功能。
- 从标准输入读取命令行参数。
- 慢发送模式，每 N 秒发送一行。
- 以十六进制显示传送或接收的数据。
- 允许其他程序服务建立连接。
- 对 Telnet 应答。

## 19.4 LSOF：隐蔽文件发现工具

当攻击者入侵 Linux 系统和上传文件后，他会设法隐藏这些文件，否则很容易被系统管理员发现。一个简单的办法就是将文件设定为隐藏。另外一种方法是创建一个进程打开一个文件，然后 unlink 文件，但是进程继续写文件。诸如 ls 之类的文件显示命令并不能显示这些信息，因此也不会被管理员发现。因此，了解系统上程序的限制是重要的。如果管理员信任一个程序，而实际上该程序并没有为使用者提供真正的信息，那么管理员会得到错误的信息。可以使用许多工具来代替标准的 ls 程序，它能够为用户提供更多信息，这个工具就是 LSOF 隐蔽文件发现工具。

该工具可以在普渡大学的主页上或者是网址：http://www.ja.net/CERT/software/lsof/上获得。LSOF 是一个能提供关于文件详细信息的程序，包括被 unlink 掉的文件。因此，当攻击者试图隐藏信息时，尽管 ls 无法发现，而 LSOF 能够帮助系统管理员发现这些文件。

## 19.5 Traceroute：路由追踪工具

Internet 是目前世界上最大的计算机网络，更确切地说是网络的网络。它由遍布全球的几万局域网和数百万台计算机组成，并通过用于异构网络的 TCP/IP 协议进行网间通信。互联网中，信息的传送是通过网中许多段的传输介质和设备（路由器、交换机、服务器、网关等）从一端到达另一端。每一个连接在 Internet 上的设备，如主机、路由器、接入服务器等一般情况下都会有一个独立的 IP 地址。通过 Traceroute 我们可以知道信息从你的计算机到互联网另一端的主机是走的什么路径。当然每次数据包由某一同样的出发点（source）到达某一同样的目的地（destination）走的路径可能会不一样，但基本上可以来说大部分时候所走的路由是相同的。UNIX 系统中，我们称之为 Traceroute、MS Windows 中为 Tracert。Traceroute 通过发送小的数据包到目的设备直到其返回，来测量其需要多长时间。一条路径上的每个设备 Traceroute 要测 3 次。输出结果中包括每次测试的时间（ms）、设备的名称（如有的话）及其 IP 地址。

Traceroute 最早是由 Van Jacobson 在 1988 写出的小程序。当时主要是为解决他自己碰到的一些网络问题。Traceroute 是一个正确理解 IP 网络并了解路由原理的重要工具。它对负责网络工程技术与系统管理的 Webmaster 是一个使用方便的程序。对 ISP 而言，设立 Traceroute 网关，将使网络服务提供商帮助用户建立并维持对服务商服务质量的信心。服务质量高的 ISP 可以通过设立 Traceroute 网关，使用户了解其与网络连接以及数据传输的效率。当然，基础设施差、服务质量低的 ISP 是比较害怕提供这种服务。因为，这样用户可以使用这一工具了解服务商目前的网络连接情况。因此，该工具可以为 Fedora 10 用户用来查看路由，并可以通过获得的信息得到报文和信息的来源，从而确定安全性。

## 19.6 XProbe：操作系统识别工具

XProbe 是一款主动操作系统指纹识别工具，它可以测定远程主机操作系统的类型。XProbe 依靠与一个签名数据库的模糊匹配以及合理的推测来确定远程操作系统的类型，利用 ICMP 协议

进行操作系统指纹识别是它的独到之处。使用时，它假设某个端口没有被使用，它会向目标主机的较高端口发送 UDP 包，目标主机就会回应 ICMP 包，然后，XProbe 会发送其他的包来分辨目标主机系统，有了这个软件，判断对方的操作系统就很容易了。

## 19.7 SATAN：系统弱点发现工具

SATAN 可以用来帮助系统管理员检测安全，也能被基于网络的攻击者用来搜索脆弱的系统。SATAN 是为系统和管理员设计的一个安全工具。然而由于它的广泛性、易用性和扫描远程网络的能力，SATAN 也可能因为好奇而被用来定位有弱点的主机。SATAN 包括一个有关网络安全问题的检测表，经过网络查找特定的系统或者子网，并报告它的发现。它能搜索以下的弱点。

- NFS：由无权限的程序或端口导出。
- NIS：口令文件访问。
- Rexd：是否被防火墙阻止。
- Sendmail：各种弱点。
- ftp：ftp、wu-ftpd 或 tftp 配置问题。
- 远程 Shell 的访问：它是否被禁止或者隐藏。
- X Windows：主机是否提供无限制的访问。
- Modem：经过 TCP 没有限制拨号访问。

# 附录 A

# 企业级 Linux 命令速查指南

作为一种优秀的操作系统，Linux 提供了 200 余种文件系统操作、系统管理和网络管理命令。因此，它受到网络管理员的青睐。这些命令能够使用户方便地使用和管理系统，更为重要的是，它们是用户安全管理 Linux 的必要知识和必备工具。基于这个原因，本附录挑选了百余种 Linux 中最为常见和重要的命令，给出的这些命令的功能说明和参数，以及语法使用都是严格地根据 Linux 中的 man 手册参考得来的，希望给用户作为日常使用和学习的参考之用。

# A.1 文件系统管理命令

## which

功能说明：查找文件。

语法：which [文件...]

使用说明：which 指令会在环境变量$PATH 设置的目录中查找符合条件的文件。

参数：

- -n<文件名长度>：指定文件名长度，指定的长度必须大于或等于所有文件中最长的文件名。
- -p<文件名长度>：与-n 参数相同，但此处的<文件名长度>包括了文件的路径。
- -w：指定输出时栏位的宽度。
- -V：显示版本信息。

## whereis

功能说明：查找文件。

语法：whereis [-bfmsu][-B<目录>...][-M<目录>...][-S<目录>...][文件...]

使用说明：whereis 指令会在特定目录中查找符合条件的文件。这些文件的属性应属于原始代码、二进制文件、或是帮助文件。

参数：

- -b：只查找二进制文件。
- -B<目录>：只在设置的目录下查找二进制文件。
- -f：不显示文件名前的路径名称。
- -m：只查找说明文件。
- -M<目录>：只在设置的目录下查找说明文件。
- -s：只查找原始代码文件。
- -S<目录>：只在设置的目录下查找原始代码文件。
- -u：查找不包含指定类型的文件。

## umask

功能说明：指定在建立文件时预设的权限掩码。

语法：umask [-S][权限掩码]

使用说明：umask 可用来设定权限掩码。权限掩码是由 3 个八进制的数字所组成，将现有的存取权限减掉权限掩码后，即可产生建立文件时预设的权限。

参数：

- -S：以文字的方式来表示权限掩码。

## split

功能说明：切割文件。

语法：split [--help][--version][-<行数>][-b<字节>][-C<字节>][-l<行数>][要切割的文件][输出

文件名]

使用说明：split 可将文件切成较小的文件，预设每 1000 行会切成一个小文件。

参数：

- -<行数>或-l<行数>：指定每多少行就要切成一个小文件。
- -b<字节>：指定每多少字就要切成一个小文件。
- -C<字节>：与-b 参数类似，但切割时尽量维持每行的完整性。
- --help：显示帮助。
- --version：显示版本信息。
- 输出文件名，设置切割后文件的前置文件名，split 会自动在前置文件名后再加上编号。

### slocate

功能说明：查找文件或目录。

语法：slocate [-u][--help][--version][-d<目录>][查找的文件]

使用说明：slocate 本身具有一个数据库，里面存放了系统中文件与目录的相关信息。

参数：

- -d<目录>或--database=<目录>：指定数据库所在的目录。
- -u：更新 slocate 数据库。
- --help：显示帮助。
- --version：显示版本信息。

### rm

功能说明：删除文件或目录。

语法：rm [-dfirv][--help][--version][文件或目录...]

使用说明：执行 rm 指令可删除文件或目录，如欲删除目录必须加上参数“-r”，否则预设仅删除文件。

参数：

- -d 或--directory：直接把欲删除的目录的硬连接数据删成 0，删除该目录。
- -f 或--force：强制删除文件或目录。
- -i 或--interactive：删除既有文件或目录之前先询问用户。
- -r 或-R 或--recursive：递归处理，将指定目录下的所有文件及子目录一并处理。
- -v 或--verbose：显示指令的执行过程。
- --help：在线帮助。
- --version：显示版本信息。

### rhmask

功能说明：产生与还原加密文件。

语法：rhmask [加密文件][输出文件] 或：rhmask [-d][加密文件][源文件][输出文件]

使用说明：执行 rhmask 指令可制作加密过的文件，方便用户在公开的网络上传输该文件，而不至于被任意盗用。

参数：

- -d：产生加密过的文件。

## rcp

功能说明：远端复制文件或目录。

语法：rcp [-pr][源文件或目录][目标文件或目录] 或：rcp [-pr][源文件或目录...][目标文件]

使用说明：rcp 指令用在远端复制文件或目录，如同时指定两个以上的文件或目录，且最后的目的地是一个已经存在的目录，则它会把前面指定的所有文件或目录复制到该目录中。

参数：

- -p：保留源文件或目录的属性，包括拥有者、所属群组、权限与时间。
- -r：递归处理，将指定目录下的文件与子目录一并处理。

## patch

功能说明：修补文件。

语法：patch [-bceEflnNRstTuvZ][-B<备份字首字符串>][-d<工作目录>][-D<标示符号>][-F<监别列数>][-g<控制数值>][-i<修补文件>][-o<输出文件>][-p<剥离层级>][-r<拒绝文件>][-V<备份方式>][-Y<备份字首字符串>][-z<备份字尾字符串>][--backup-if：-mismatch][--binary][--help][--nobackup-if-mismatch][--verbose][原始文件<修补文件>]

或：path [-p<剥离层级>]< [修补文件]

使用说明：patch 指令让用户利用设置修补文件的方式，修改、更新原始文件。倘若一次仅修改一个文件，可直接在指令列中下达指令依序执行。如果配合修补文件的方式则能一次修补大批文件，这也是 Linux 系统核心的升级方法之一。

参数：

- -b 或--backup：备份每一个原始文件。
- -B<备份字首字符串>或--prefix=<备份字首字符串>：设置文件备份时，附加在文件名称前面的字首字符串，该字符串可以是路径名称。
- -c 或--context：把修补数据编译成关联性的差异。
- -d<工作目录>或--directory=<工作目录>：设置工作目录。
- -D<标示符号>或--ifdef=<标示符号>：用指定的符号把改变的地方标示出来。
- -e 或--ed：把修补数据解译成 ed 指令可用的叙述文件。
- -E 或--remove-empty-files：若修补过后输出的文件其内容是一片空白，则移除该文件。
- -f 或--force：此参数的效果和指定“-t”参数类似，但会假设修补数据的版本为新版本。
- -F<监别列数>或--fuzz<监别列数>：设置监别列数的最大值。
- -g<控制数值>或--get=<控制数值>：设置以 RSC 或 SCCS 控制修补文件。
- -i<修补文件>或--input=<修补文件>：读取指定的修补文件。
- -l 或--ignore-whitespace：忽略修补数据与输入数据的空格字符。
- -n 或--normal：把修补数据编译成一般性的差异。
- -N 或--forward：忽略修补的数据较原始文件的版本更旧，或该版本的修补数据已使用过。
- -o<输出文件>或--output=<输出文件>：设置输出文件的名称，修补过的文件会以该名称

存放。

- -p<剥离层级>或--strip=<剥离层级>：设置欲剥离几层路径名称。
- -f<拒绝文件>或--reject-file=<拒绝文件>：设置保存拒绝修补相关信息的文件名称，预设的文件名称为.rej。
- -R 或--reverse：假设修补数据是由新旧文件交换位置而产生。
- -s 或--quiet 或--silent：不显示指令执行过程，除非发生错误。
- -t 或--batch：自动略过错误，不询问任何问题。
- -T 或--set-time：此参数的效果和指定“-Z”参数类似，但以本地时间为主。
- -u 或--unified：把修补数据编译成一致化的差异。
- -v 或--version：显示版本信息。
- -V<备份方式>或--version-control=<备份方式>：用“-b”参数备份目标文件后，备份文件的字尾会被加上一个备份字符串，这个字符串不仅可用“-z”参数变更，当使用“-V”参数指定不同备份方式时，也会产生不同字尾的备份字符串。
- -Y<备份字首字符串>或--basename-prefix=--<备份字首字符串>：设置文件备份时，附加在文件基本名称开头的字首字符串。
- -z<备份字尾字符串>或--suffix=<备份字尾字符串>：此参数的效果和指定"-B"参数类似，差别在于修补作业使用的路径与文件名若为 src/linux/fs/super.c，加上"backup/"字符串后，文件 super.c 会备份于/src/linux/fs/backup 目录中。
- -Z 或--set-utc：把修补过的文件更改，存取时间设为 UTC。
- --backup-if-mismatch：在修补数据不完全吻合，且没有刻意指定要备份文件时，才备份文件。
- --binary：以二进制模式读写数据，而不通过标准输出设备。
- --help：在线帮助。
- --nobackup-if-mismatch：在修补数据不完全吻合，且没有刻意指定要备份文件时，不要备份文件。
- --verbose：详细显示指令的执行过程。

## paste

功能说明：合并文件的列。

语法：paste [-s][-d<间隔字符>][--help][--version][文件...]

使用说明：paste 指令会把每个文件以列对列的方式，一列列地加以合并。

参数：

- -d<间隔字符>或--delimiters=<间隔字符>：用指定的间隔字符取代跳格字符。
- -s 或--serial：串列进行而非平行处理。
- --help：在线帮助。
- --version：显示帮助信息。

## od

功能说明：输出文件内容。

语法：od [-abcdfhilovx][-A<字码基数>][-j<字符数目>][-N<字符数目>][-s<字符串字符

数>][-t<输出格式>][-w<每列字符数>][--help][--version][文件...]

使用说明：od 指令会读取所给予的文件的内容，并将其内容以八进制字码呈现出来。

参数：

- -a：此参数的效果和同时指定“-ta”参数相同。
- -A<字码基数>：选择要以何种基数计算字码。
- -b：此参数的效果和同时指定“-toC”参数相同。
- -c：此参数的效果和同时指定“-tC”参数相同。
- -d：此参数的效果和同时指定“-tu2”参数相同。
- -f：此参数的效果和同时指定“-tfF”参数相同。
- -h：此参数的效果和同时指定“-tx2”参数相同。
- -i：此参数的效果和同时指定“-td2”参数相同。
- -j<字符数目>或--skip-bytes=<字符数目>：略过设置的字符数目。
- -l：此参数的效果和同时指定“-td4”参数相同。
- -N<字符数目>或--read-bytes=<字符数目>：到设置的字符数目为止。
- -o：此参数的效果和同时指定“-to2”参数相同。
- -s<字符串字符数>或--strings=<字符串字符数>：只显示符合指定的字符数目的字符串。
- -t<输出格式>或--format=<输出格式>：设置输出格式。
- -v 或--output-duplicates：输出时不省略重复的数据。
- -w<每列字符数>或--width=<每列字符数>：设置每列的最大字符数。
- -x：此参数的效果和同时指定“-h”参数相同。
- --help：在线帮助。
- --version：显示版本信息。

## mv

功能说明：移动或更名现有的文件或目录。

语法：mv [-bfiuv][--help][--version][-S<附加字尾>][-V<方法>][源文件或目录][目标文件或目录]

使用说明：mv 指令可移动文件或目录，或是更改文件或目录的名称。

参数：

- -b 或--backup：若需覆盖文件，则覆盖前先进行备份。
- -f 或--force：若目标文件或目录与现有的文件或目录重复，则直接覆盖现有的文件或目录。
- -i 或--interactive：覆盖前先询问用户。
- -S<附加字尾>或--suffix=<附加字尾>：与-b 参数一并使用，可指定备份文件的所要附加的字尾。
- -u 或--update：在移动或更改文件名时，若目标文件已存在，且其文件日期比源文件新，则不覆盖目标文件。
- -v 或--verbose：执行时显示详细的信息。
- -V=<方法>或--version-control=<方法>：与-b 参数一并使用，可指定备份的方法。
- --help：显示帮助。
- --version：显示版本信息。

## mktemp

功能说明：建立暂存文件。

语法：mktemp [-qu][文件名参数]

使用说明：mktemp 指令可建立一个暂存文件，供 shellscript 使用。

参数：

- -q：执行时若发生错误，不会显示任何信息。
- -u：暂存文件会在 mktemp 结束前先删除。
- 文件名参数：文件名参数必须是以“自定名称.XXXXXX”的格式。

## lsattr

功能说明：显示文件属性。

语法：lsattr [-adlRvV][文件或目录...]

使用说明：用 chattr 执行改变文件或目录的属性，可执行 lsattr 指令查询其属性。

参数：

- -a：显示所有文件和目录，包括以“.”为名称开头字符的额外内建、现行目录“.”与上层目录“..”。
- -d：显示目录名称，而非其内容。
- -l：此参数目前没有任何作用。
- -R：递归处理，将指定目录下的所有文件及子目录一并处理。
- -v：显示文件或目录版本。
- -V：显示版本信息。

## locate

功能说明：查找文件。

语法：locate [-d<数据库文件>][--help][--version][范本样式...]

使用说明：locate 指令用于查找符合条件的文件，它会在保存文件与目录名称的数据库中，查找合乎范本样式条件的文件或目录。

参数：

- -d<数据库文件>或--database=<数据库文件>：设置 locate 指令使用的数据库。locate 指令预设的数据库位于/var/lib/slocate 目录中，文件名为 slocate.db，您可使用该参数另行指定。
- --help：在线帮助。
- --version：显示版本信息。

## file

功能说明：辨识文件类型。

语法：file [-beLvz][-f<名称文件>][-m<魔法数字文件>...][文件或目录...]

使用说明：通过 file 指令，我们得以辨识该文件的类型。

参数：

- -b：列出辨识结果时，不显示文件名称。
- -c：详细显示指令的执行过程，便于排错或分析程序执行的情形。
- -f<名称文件>：指定名称文件，其内容有一个或多个文件名称，让 file 依序辨识这些文件，格式为每列一个文件名称。
- -L：直接显示符号连接所指向的文件的类别。
- -m<魔法数字文件>：指定魔法数字文件。
- -v：显示版本信息。
- -z：尝试去解读压缩文件的内容。

## diffstat

功能说明：根据 diff 的比较结果，显示统计数字。

语法：diff [-wV][-n<文件名长度>][-p<文件名长度>]

使用说明：diffstat 读取 diff 的输出结果，然后统计各文件的插入、删除、修改等差异计量。

参数：

- -n<文件名长度>：指定文件名长度，指定的长度必须大于或等于所有文件中最长的文件名。
- -p<文件名长度>：与-n 参数相同，但此处的“文件名长度”包括了文件的路径。
- -w：指定输出时栏位的宽度。
- -V：显示版本信息。

## cat

功能说明：显示文件内容。

语法：cat [-AbeEnstTuv][--help][--version]fileName：

使用说明：把档案串连接后传到基本输出（屏幕或加 fileName 输出到另一个档案）。

参数：

- -n 或--number 由：1 开始对所有输出的行数编号。
- -b 或--number-nonblank：和-n 相似，只不过对于空白行不编号。
- -s 或--squeeze-blank：当遇到有连续两行以上的空白行，就替换为一行的空白行。
- -v 或--show-nonprinting：显示版本信息。

## chattr

功能说明：改变文件属性。

语法：chattr [-RV][-v<版本编号>][+/-/=<属性>][文件或目录...]

使用说明：chattr 指令可改变存放在 ext2 文件系统上的文件或目录属性，这些属性共有以下 8 种模式。

- a：让文件或目录仅供附加用途。
- b：不更新文件或目录的最后存取时间。
- c：将文件或目录压缩后存放。
- d：将文件或目录排除在倾倒操作之外。
- i：不得任意改动文件或目录。
- s：保密性删除文件或目录。

- S：即时更新文件或目录。
- u：预防意外删除。

参数：

- -R：递归处理，将指定目录下的所有文件及子目录一并处理。
- -v<版本编号>：设置文件或目录版本。
- -V：显示指令执行过程。
- +<属性>：开启文件或目录的该项属性。
- -<属性>：关闭文件或目录的该项属性。
- =<属性>：指定文件或目录的该项属性。

## chgrp

功能说明：变更文件或目录的所属群组。

语法：chgrp [-cfhRv][--help][--version][所属群组][文件或目录...]

或：chgrp [-cfhRv][--help][--reference=<参考文件或目录>][--version][文件或目录...]

使用说明：在UNIX系统家族里，文件或目录权限的掌控以拥有者及所属群组来管理。您可以使用chgrp指令去变更文件与目录的所属群组，设置方式采用群组名称或群组识别码均可。

参数：

- -c或--changes：效果类似“-v”参数，但仅返回更改的部分。
- -f或--quiet或--silent：不显示错误信息。
- -h或--no-dereference：只对符号连接的文件作修改，而不改动其他任何相关文件。
- -R或--recursive：递归处理，将指定目录下的所有文件及子目录一并处理。
- -v或--verbose：显示指令的执行过程。
- --help：在线帮助。
- --reference=<参考文件或目录>：把指定文件或目录的所属群组全部设成和参考文件或目录的所属群组相同。
- --version：显示版本信息。

## find

功能说明：查找文件或目录。

语法：find [目录...][-amin<分钟>][-anewer<参考文件或目录>][-attime<24小时数>][-cmin<分钟>][-cnewer<参考文件或目录>][-ctime<24小时数>][-daystart][-depth][-empty][-exec<执行指令>][-false][-fls<列表文件>][-follow][-fprint<列表文件>][-fprint0<列表文件>][-fprintf<列表文件><输出格式>][-fstype<文件系统类型>][-gid<群组识别码>][-group<群组名称>][-help][-ilname<范本样式>][-iname<范本样式>][-inum<inode编号>][-ipath<范本样式>][-iregex<范本样式>][-links<连接数目>][-lname<范本样式>][-ls][-maxdepth<目录层级>][-mindepth<目录层级>][-mmin<分钟>][-mount][-mtime<24小时数>][-name<范本样式>][-newer<参考文件或目录>][-nogroup][noleaf][-nouser][-ok<执行指令>][-path<范本样式>][-perm<权限数值>][-print][-print0][-printf<输出格式>][-prune][-regex<范本样式>][-size<文件大小>][-true][-type<文件类型>][-uid<用户识别码>][-used<日数>][-user<拥有者名称>][-version][-xdev][-xtype<文件类型>]

使用说明：find 指令用于查找符合条件的文件。任何位于参数之前的字符串都将被视为欲查找的目录。

参数：

- -amin<分钟>：查找在指定时间曾被存取过的文件或目录，单位以分钟计算。
- -anewer<参考文件或目录>：查找其存取时间较指定文件或目录的存取时间更接近现在的文件或目录。
- -atime<24 小时数>：查找在指定时间曾被存取过的文件或目录，单位以 24 小时计算。
- -cmin<分钟>：查找在指定时间之时被更改的文件或目录。
- -cnewer<参考文件或目录>：查找其更改时间较指定文件或目录的更改时间更接近现在的文件或目录。
- -ctime<24 小时数>：查找在指定时间之时被更改的文件或目录，单位以 24 小时计算。
- -daystart：从本日开始计算时间。
- -depth：从指定目录下最深层的子目录开始查找。
- -empty：寻找文件大小为 0Byte 的文件，或目录下没有任何子目录或文件的空目录。
- -exec<执行指令>：假设 find 指令的回传值为 True，则执行该指令。
- -false：将 find 指令的回传值皆设为 False。
- -fls<列表文件>：此参数的效果和指定“-ls”参数类似，但会把结果保存为指定的列表文件。
- -follow：排除符号连接。
- -fprint<列表文件>：此参数的效果和指定“-print”参数类似，但会把结果保存成指定的列表文件。
- -fprint0<列表文件>：此参数的效果和指定“-print0”参数类似，但会把结果保存成指定的列表文件。
- -fprintf<列表文件><输出格式>：此参数的效果和指定“-printf”参数类似，但会把结果保存成指定的列表文件。
- -fstype<文件系统类型>：只寻找该文件系统类型下的文件或目录。
- -gid<群组识别码>：查找符合指定的群组识别码的文件或目录。
- -group<群组名称>：查找符合指定的群组名称的文件或目录。
- -help 或--help：在线帮助。
- -ilname<范本样式>：此参数的效果和指定“-lname”参数类似，但忽略字符大小写的差别。
- -iname<范本样式>：此参数的效果和指定“-name”参数类似，但忽略字符大小写的差别。
- -inum<inode 编号>：查找符合指定的 inode 编号的文件或目录。
- -ipath<范本样式>：此参数的效果和指定“-ipath”参数类似，但忽略字符大小写的差别。
- -iregex<范本样式>：此参数的效果和指定“-regexe”参数类似，但忽略字符大小写的差别。
- -links<连接数目>：查找符合指定的硬连接数目的文件或目录。
- -lname<范本样式>：指定字符串作为寻找符号连接的范本样式。
- -ls：假设 find 指令的回传值为 True，就将文件或目录名称列出到标准输出。
- -maxdepth<目录层级>：设置最大目录层级。
- -mindepth<目录层级>：设置最小目录层级。
- -mmin<分钟>：查找在指定时间曾被更改过的文件或目录，单位以分钟计算。

- -mount：此参数的效果和指定“-xdev”相同。
- -mtime<24 小时数>：查找在指定时间曾被更改过的文件或目录，单位以 24 小时计算。
- -name<范本样式>：指定字符串作为寻找文件或目录的范本样式。
- -newer<参考文件或目录>：查找其更改时间较指定文件或目录的更改时间更接近现在的文件或目录。
- -nogroup：找出不属于本地主机群组识别码的文件或目录。
- -noleaf：不去考虑目录至少需拥有两个硬连接存在。
- -nouser：找出不属于本地主机用户识别码的文件或目录。
- -ok<执行指令>：此参数的效果和指定“-exec”参数类似，但在执行指令之前会先询问用户，若回答“y”或“Y”，则放弃执行指令。
- -path<范本样式>：指定字符串作为寻找目录的范本样式。
- -perm<权限数值>：查找符合指定的权限数值的文件或目录。
- -print：假设 find 指令的回传值为 True，则将文件或目录名称列出到标准输出。格式为每列一个名称，每个名称之前皆有“./”字符串。
- -print0：假设 find 指令的回传值为 True，则将文件或目录名称列出到标准输出。格式为全部的名称均在同一行。
- -printf<输出格式>：假设 find 指令的回传值为 True，则将文件或目录名称列出到标准输出。格式可以自行指定。
- -prune：不寻找字符串作为寻找文件或目录的范本样式。
- -regex<范本样式>：指定字符串作为寻找文件或目录的范本样式。
- -size<文件大小>：查找符合指定的文件大小的文件。
- -true：将 find 指令的回传值皆设为 True。
- -typ<文件类型>：只寻找符合指定的文件类型的文件。
- -uid<用户识别码>：查找符合指定的用户识别码的文件或目录。
- -used<日数>：查找文件或目录被更改之后在指定时间曾被存取过的文件或目录，单位以日计算。
- -user<拥有者名称>：查找符合指定的拥有者名称的文件或目录。
- -version 或--version：显示版本信息。
- -xdev：将范围局限在先行的文件系统中。
- -xtype<文件类型>：此参数的效果和指定“-type”参数类似，差别在于它针对符号连接检查。

## cksum

功能说明：检查文件的 CRC 是否正确。

语法：cksum [--help][--version][文件...]

使用说明：CRC 是一种排错检查方式，该演算法的标准由 CCITT 所指定，至少可检测到 99.998%的已知错误。指定文件交由 cksum 演算，它会回报计算结果，供用户核对文件是否正确无误。若不指定任何文件名称或是所给予的文件名为“-”，则 cksum 指令会从标准输入设备读取数据。

参数：

- --help：在线帮助。
- --version：显示版本信息。

## gitview

功能说明：Hex/ASCII 的观看文件程序。

语法：gitview [-bchilv][文件]

使用说明：gitview 指令可用于观看文件的内容，它会同时显示十六进制和 ASCII 格式的字码。

参数：

- -b：单色模式，不使用 ANSI 控制码显示彩色。
- -c：彩色模式，使用 ANSI 控制码显示色彩。
- -h：在线帮助。
- -i：显示存放 gitview 程序的所在位置。
- -l：不使用先前的显示字符。
- -v：显示版本信息。

## cmp

功能说明：比较两个文件是否有差异。

语法：cmp [-clsv][-i<字符数目>][--help][第一个文件][第二个文件]

使用说明：当相互比较的两个文件完全一样时，则该指令不会显示任何信息。若发现有所差异，预设会标示出第一个不同之处的字符和列数编号。若不指定任何文件名称或是所给予的文件名为“-”，则 cmp 指令会从标准输入设备读取数据。

参数：

- -c 或--print-chars：除了标明差异处的十进制字码之外，一并显示该字符所对应的字符。
- -i<字符数目>或--ignore-initial=<字符数目>：指定一个数目。
- -l 或--verbose：标示出所有不一样的地方。
- -s 或--quiet 或--silent：不显示错误信息。
- -v 或--version：显示版本信息。
- --help：在线帮助。

## indent

功能说明：调整 C 原始代码文件的格式。

语法：indent [参数][源文件] 或：indent [参数][源文件][-o：目标文件]

使用说明：indent 可辨识 C 的原始代码文件，并加以格式化，以方便程序设计师阅读。

参数：

- -bad 或--blank-lines-after-declarations：在声明区段后加上空白行。
- -bap 或--blank-lines-after-procedures：在程序后加上空白行。
- -bbb 或--blank-lines-after-block-comments：在注释区段后加上空白行。
- -bc 或--blank-lines-after-commas：在声明区段中，若出现逗号即换行。
- -bl 或--braces-after-if-line：if（或是 else、for 等）与后面执行区段的“{”不同行，且“}”

自成一行。

- -bli<缩排格数>或--brace-indent<缩排格数>：设置“{”“}”缩排的格数。
- -br 或--braces-on-if-line：if（或是 else、for 等）与后面执行区段的“{”不同行，且“}”自成一行。
- -bs 或--blank-before-sizeof：在 sizeof 之后空一格。
- -c<栏数>或--comment-indentation<栏数>：将注释置于程序码右侧指定的栏位。
- -cd<栏数>或--declaration-comment-column<栏数>：将注释置于声明右侧指定的栏位。
- -cdb 或--comment-delimiters-on-blank-lines：注释符号自成一行。
- -ce 或--cuddle-else：将 else 置于“}”（if 执行区段的结尾）之后。
- -ci<缩排格数>或--continuation-indentation<缩排格数>：叙述过长而换行时，指定换行后缩排的格数。
- -cli<缩排格数>或--case-indentation-<缩排格数>：使用 case 时，switch 缩排的格数。
- -cp<栏数>或-else-endif-column<栏数>：将注释置于 else 与 elseif 叙述右侧指定的栏位。
- -cs 或--space-after-cast：在 cast 之后空一格。
- -d<缩排格数>或-line-comments-indentation<缩排格数>：针对不是放在程序码右侧的注释，设置其缩排格数。
- -di<栏数>或--declaration-indentation<栏数>：将声明区段的变量置于指定的栏位。
- -fc1 或--format-first-column-comments：针对放在每行最前端的注释，设置其格式。
- -fca 或--format-all-comments：设置所有注释的格式。
- -gnu 或--gnu-style：指定使用 GNU 的格式，此为预设值。
- -i<格数>或--indent-level<格数>：设置缩排的格数。
- -ip<格数>或--parameter-indentation<格数>：设置参数的缩排格数。
- -kr 或--k-and-r-style：指定使用 Kernighan&Ritchie 的格式。
- -lp 或--continue-at-parentheses：叙述过长而换行，且叙述中包含了括弧时，将括弧中的每行起始栏位内容垂直对齐排列。
- -nbad 或--no-blank-lines-after-declarations：在声明区段后不要加上空白行。
- -nbap 或--no-blank-lines-after-procedures：在程序后不要加上空白行。
- -nbbb 或--no-blank-lines-after-block-comments：在注释区段后不要加上空白行。
- -nbc 或--no-blank-lines-after-commas：在声明区段中，即使出现逗号，仍旧不要换行。
- -ncdb 或--no-comment-delimiters-on-blank-lines：注释符号不要自成一行。
- -nce 或--dont-cuddle-else：不要将 else 置于“}”之后。
- -ncs 或--no-space-after-casts：不要在 cast 之后空一格。
- -nfc1 或--dont-format-first-column-comments：不要格式化放在每行最前端的注释。
- -nfca 或--dont-format-comments：不要格式化任何的注释。
- -nip 或--no-parameter-indentation：参数不要缩排。
- -nlp 或--dont-line-up-parentheses：叙述过长而换行，且叙述中包含了括弧时，不用将括弧中的每行起始栏位垂直对齐排列。
- -npcs 或--no-space-after-function-call-names：在调用的函数名称之后，不要加上空格。
- -npro 或--ignore-profile：不要读取 indent 的配置文件 indent.pro。
- -npsl 或--dont-break-procedure-type：程序类型与程序名称放在同一行。

- -nsc 或--dont-star-comments：注解左侧不要加上星号（*）。
- -nsob 或--leave-optional-semicolon：不用处理多余的空白行。
- -nss 或--dont-space-special-semicolon：若 for 或 while 区段仅有一行时，在分号前不加上空格。
- -nv 或--no-verbosity：不显示详细的信息。
- -orig 或--original：使用 Berkeley 的格式。
- -pcs 或--space-after-procedure-calls：在调用的函数名称与"{"之间加上空格。
- -psl 或--procnames-start-lines：程序类型置于程序名称的前一行。
- -sc 或--start-left-side-of-comments：在每行注释左侧加上星号（*）。
- -sob 或--swallow-optional-blank-lines：删除多余的空白行。
- -ss 或--space-special-semicolon：若 for 或 swile 区段仅有一行时，在分号前加上空格。
- -st 或--standard-output：将结果显示在标准输出设备。
- -T：数据类型名称缩排。
- -ts<格数>或--tab-size<格数>：设置 tab 的长度。
- -v 或--verbose：执行时显示详细的信息。
- -version：显示版本信息。

## cp

功能说明：复制文件或目录。

语法：cp [-abdfilpPrRsuvx][-S<备份字尾字符串>][-V<备份方式>][--help][--spares=<使用时机>][--version][源文件或目录][目标文件或目录][目的目录]

使用说明：cp 指令用在复制文件或目录，如同时指定两个以上的文件或目录，且最后的目的地是一个已经存在的目录，则它会把前面指定的所有文件或目录复制到该目录中。若同时指定多个文件或目录，而最后的目的地并非是一个已存在的目录，则会出现错误信息。

参数：

- -a 或--archive：此参数的效果和同时指定"-dpR"参数相同。
- -b 或--backup：删除、覆盖目标文件之前的备份，备份文件会在字尾加上一个备份字符串。
- -d 或--no-dereference：当复制符号连接时，把目标文件或目录也建立为符号连接，并指向与源文件或目录连接的原始文件或目录。
- -f 或--force：强行复制文件或目录，不论目标文件或目录是否已存在。
- -i 或--interactive：覆盖既有文件之前先询问用户。
- -l 或--link：对源文件建立硬连接，而非复制文件。
- -p 或--preserve：保留源文件或目录的属性。
- -P 或--parents：保留源文件或目录的路径。
- -r：递归处理，将指定目录下的文件与子目录一并处理。
- -R 或--recursive：递归处理，将指定目录下的所有文件与子目录一并处理。
- -s 或--symbolic-link：对源文件建立符号连接，而非复制文件。
- -S<备份字尾字符串>或--suffix=<备份字尾字符串>：用"-b"参数备份目标文件后，备份文件的字尾会被加上一个备份字符串，预设的备份字尾字符串是符号"~"。
- -u 或--update：使用该参数后只会在源文件的更改时间较目标文件更新时或是名称相互

对应的目标文件并不存在时，才复制文件。

- -v 或--verbose：显示指令的执行过程。
- -V<备份方式>或--version-control=<备份方式>：用“-b”参数备份目标文件后，备份文件的字尾会被加上一个备份字符串，该字符串不仅可用“-S”参数变更，当使用“-V”参数指定不同备份方式时，也会产生不同字尾的备份字串。
- -x 或--one-file-system：复制的文件或目录存放的文件系统，必须与 cp 指令执行时所处的文件系统相同，否则不予复制。
- --help：在线帮助。
- --sparse=<使用时机>：设置保存稀疏文件的时机。
- --version：显示版本信息。

## touch

功能说明：改变文件或目录时间。

语法：touch [-acfm][-d<日期时间>][-r<参考文件或目录>][-t<日期时间>][--help] [--version][文件或目录...]

或：touch [-acfm][--help][--version][日期时间][文件或目录...]

使用说明：使用 touch 指令可更改文件或目录的日期时间，包括存取时间和更改时间。

参数：

- -a 或--time=atime 或--time=access 或--time=use：只更改存取时间。
- -c 或--no-create：不建立任何文件。
- -d<日期时间>：使用指定的日期时间，而非现在的时间。
- -f：此参数将忽略不予处理，仅负责解决 BSD 版本 touch 指令的兼容性问题。
- -m 或--time=mtime 或--time=modify：只更改变动时间。
- -r<参考文件或目录>：把指定文件或目录的日期时间全部设成和参考文件或目录的日期时间相同。
- -t<日期时间>：使用指定的日期时间，而非现在的时间。
- --help：在线帮助。
- --version：显示版本信息。

## Tmpwatch

功能说明：删除暂存文件。

语法：tmpwatch [-afqv][--test][超期时间][目录...]

使用说明：执行 tmpwatch 指令可删除不必要的暂存文件，您可以设置文件的超期时间，单位以小时计算。

参数：

- -a 或--all：删除任何类型的文件。
- -f 或--force：强制删除文件或目录，其效果类似 rm 指令的“-f”参数。
- -q 或--quiet：不显示指令执行过程。
- -v 或--verbose：详细显示指令执行过程。
- -test：仅作测试，并不真地删除文件或目录。

## tee

功能说明：读取标准输入的数据，并将其内容输出成文件。

语法：tee [-ai][--help][--version][文件...]

使用说明：tee 指令会从标准输入设备读取数据，将其内容输出到标准输出设备，同时保存成文件。

参数：

- -a 或--append：附加到既有文件的后面，而非覆盖它。
- -i-i 或--ignore-interrupts：忽略中断信号。
- --help：在线帮助。
- --version：显示版本信息。

## cut

使用权限：所有使用者。

用法：cut-cnum1-num2：filename

说明：显示每行从开头算起 num1～num2：的文字。

## diff

功能说明：比较文件的差异。

语法：diff [-abBcdefHilnNpPqrstTuvwy][-<行数>][-C<行数>][-D<巨集名称>][-I<字符或字符串>][-S<文件>][-W<宽度>][-x<文件或目录>][-X<文件>][--help][--left-column][--suppress-common-line][文件或目录 1][文件或目录 2]

使用说明：diff 以逐行的方式，比较文本文件的异同处。指定要比较的目录，则 diff 会比较目录中相同文件名的文件，但不会比较其中子目录。

参数：

- -<行数>：指定要显示多少行的文本。此参数必须与-c 或-u 参数一并使用。
- -a 或--text：diff 预设只会逐行比较文本文件。
- -b 或--ignore-space-change：不检查空格字符的不同。
- -B 或--ignore-blank-lines：不检查空白行。
- -c：显示全部内文，并标出不同之处。
- -C<行数>或--context<行数>：与执行“-c-<行数>”指令相同。
- -d 或--minimal：使用不同的演算法，以较小的单位来做比较。
- -D<巨集名称>或 ifdef<巨集名称>：此参数的输出格式可用于前置处理器巨集。
- -e 或--ed：此参数的输出格式可用于 ed 的 script 文件。
- -f 或-forward-ed：输出的格式类似 ed 的 script 文件，但按照原来文件的顺序来显示不同处。
- -H 或--speed-large-files：比较大文件时，可加快速度。
- -l<字符或字符串>或--ignore-matching-lines<字符或字符串>：若两个文件在某几行有所不同，而这几行同时都包含了选项中指定的字符或字符串，则不显示这两个文件的差异。
- -i 或--ignore-case：不检查大小写的不同。
- -l 或--paginate：将结果交由 pr 程序来分页。

- -n 或--rcs：将比较结果以 RCS 的格式来显示。
- -N 或--new-file：在比较目录时，若文件 A 仅出现在某个目录中，预设会显示。
- Only：in 目录：文件 A 若使用-N 参数，则 diff 会将文件 A 与一个空白的文件比较。
- -p：若比较的文件为 C 语言的程序码文件时，显示差异所在的函数名称。
- -P 或--unidirectional-new-file：与-N 类似，但只有当第二个目录包含了一个第一个目录所没有的文件时，才会将这个文件与空白的文件做比较。
- -q 或--brief：仅显示有无差异，不显示详细的信息。
- -r 或--recursive：比较子目录中的文件。
- -s 或--report-identical-files：若没有发现任何差异，仍然显示信息。
- -S<文件>或--starting-file<文件>：在比较目录时，从指定的文件开始比较。
- -t 或--expand-tabs：在输出时，将 tab 字符展开。
- -T 或--initial-tab：在每行前面加上 tab 字符以便对齐。
- -u,-U<列数>或--unified=<列数>：以合并的方式来显示文件内容的不同。
- -v 或--version：显示版本信息。
- -w 或--ignore-all-space：忽略全部的空格字符。
- -W<宽度>或--width<宽度>：在使用-y 参数时，指定栏宽。
- -x<文件或目录>或--exclude<文件或目录>：不比较选项中所指定的文件或目录。
- -X<文件>或--exclude-from<文件>：您可以将文件或目录类型存成文本文件，然后在 =<文件>中指定此文本文件。
- -y 或--side-by-side：以并列的方式显示文件的异同之处。
- --help：显示帮助。
- --left-column：在使用-y 参数时，若两个文件某一行内容相同，则仅在左侧的栏位显示该行内容。
- --suppress-common-line：在使用-y 参数时，仅显示不同之处。

## A.2 系统管理命令

### groupdel

功能说明：删除群组。

语法：groupdel [群组名称]

使用说明：需要从系统上删除群组时，可用 groupdel 指令来完成这项工作。如果该群组中仍包括某些用户，则必须先删除这些用户，能删除群组。

### groupmod

功能说明：更改群组识别码或名称。

语法：groupmod [-g<群组识别码><-o>][-n<新群组名称>][群组名称]

使用说明：需要更改群组的识别码或名称时，可用 groupmod 指令来完成这项工作。

参数：

- -g<群组识别码>：设置欲使用的群组识别码。
- -o：重复使用群组识别码。
- -n<新群组名称>：设置欲使用的群组名称。

## id

功能说明：显示用户的 ID，以及所属群组的 ID。

语法：id [-gGnru][--help][--version][用户名称]

使用说明：id 会显示用户以及所属群组的实际与有效 ID。若两个 ID 相同，则仅显示实际 ID。若仅指定用户名称，则显示目前用户的 ID。

参数：

- -g 或--group：显示用户所属群组的 ID。
- -G 或--groups：显示用户所属附加群组的 ID。
- -n 或--name：显示用户、所属群组或附加群组的名称。
- -r 或--real：显示实际 ID。
- -u 或--user：显示用户 ID。
- -help：显示帮助。
- -version：显示版本信息。

## pstree

功能说明：以树状图显示程序。

语法：pstree [-acGhlnpuUV][-H<程序识别码>][<程序识别码>/<用户名称>]

使用说明：pstree 指令用 ASCII 字符显示树状结构，清楚地表达程序间的相互关系。如果不指定程序识别码或用户名称，则会把系统启动时的第一个程序视为基层，并显示之后的所有程序。若指定用户名称，便会以隶属该用户的第一个程序当作基层，然后显示该用户的所有程序。

参数：

- -a：显示每个程序的完整指令，包含路径、参数或是常驻服务的标示。
- -c：不使用精简标示法。
- -G：使用 VT100 终端机的列绘图字符。
- -h：列出树状图时，特别标明现在执行的程序。
- -H<程序识别码>：此参数的效果和指定“-h”参数类似，但特别标明指定的程序。
- -l：采用长列格式显示树状图。
- -n：用程序识别码排序。预设是以程序名称来排序。
- -p：显示程序识别码。
- -u：显示用户名称。
- -U：使用 UTF-8 列绘图字符。
- -V：显示版本信息。

## rlogin

功能说明：远端登录。

语法：rlogin [-8EL][-e<脱离字符>][-l<用户名称>][主机名称或 IP 地址]

使用说明：执行 rlogin 指令开启终端机阶段操作，并登录远端主机。

参数：

- -8：允许输入 8 位字符数据。
- -e<脱离字符>：设置脱离字符。
- -E：滤除脱离字符。
- -l<用户名称>：指定要登录远端主机的用户名称。
- -L：使用 litout 模式进行远端登录阶段操作。

## rsh

功能说明：远端登录的 shell。

语法：rsh [-dn][-l<用户名称>][主机名称或 IP 地址][执行指令]

使用说明：rsh 提供用户环境，也就是 shell，以便指令能够在指定的远端主机上执行。

参数：

- -d：使用 Socket 层级的排错功能。
- -l<用户名称>：指定要登录远端主机的用户名称。
- -n：把输入的指令号指向代号为/dev/null 的特殊外围设备。

## rwho

功能说明：查看系统用户。

语法：rwho [-a]

使用说明：rwho 指令的效果类似 who 指令，但它会显示局域网里所有主机的用户。主机必须提供 rwhod 常驻服务的功能，方可使用 rwho 指令。

参数：

- -a：列出所有的用户，包括闲置时间超过 1 个小时以上的用户。

## lsmod

功能说明：显示已载入系统的模块。

语法：lsmod

使用说明：执行 lsmod 指令，会列出所有已载入系统的模块。Linux 操作系统的核心具有模块化的特性，因此在编译核心时，必须把全部的功能都放入核心。您可以将这些功能编译成一个个单独的模块，待需要时再分别载入。

## insmod

功能说明：载入模块。

语法：insmod [-fkmpsvxX][-o<模块名称>][模块文件][符号名称=符号值]

使用说明：Linux 有许多功能是通过模块的方式，在需要时才载入 kernel。如此可使 kernel 较为精简，进而提高效率，以及保有较大的弹性。这类可载入的模块，通常是设备驱动程序。

参数：

- -f：不检查目前 kernel 版本与模块编译时的 kernel 版本是否一致，强制将模块载入。
- -k：将模块设置为自动卸载。

- -m：输出模块的载入信息。
- -o<模块名称>：指定模块的名称，可使用模块文件的文件名。
- -p：测试模块是否能正确地载入 kernel。
- -s：将所有信息记录在系统记录文件中。
- -v：执行时显示详细的信息。
- -x：不要输出模块的外部符号。
- -X：输出模块所有的外部符号，此为预设置。

## grpunconv

功能说明：关闭群组的投影密码。

语法：grpunconv

使用说明：执行 grpunconv 指令可关闭群组投影密码，它会把密码从 gshadow 文件内，改存到 group 文件里。

## grpconv

功能说明：开启群组的投影密码。

语法：grpconv

使用说明：Linux 系统里的用户和群组密码，分别存放在/etc 目录下的 passwd 和 group 文件中。因系统运作所需，任何人都得以读取它们，造成安全上的破绽。投影密码将文件内的密码改存在/etc 目录下的 shadow 和 gshadow 文件内，只允许系统管理者读取，同时把原密码置换为“x”字符。投影密码的功能可随时开启或关闭，您只需执行 grpconv 指令就能开启群组投影密码。

## export

功能说明：设置或显示环境变量。

语法：export [-fnp][变量名称]=[变量设置值]

使用说明：在 shell 中执行程序时，shell 会提供一组环境变量。export 可新增、修改或删除环境变量，供后续执行的程序使用。export 的效力仅及于该此登录操作。

参数：

- -f：代表“变量名称”中为函数名称。
- -n：删除指定的变量。变量实际上并未删除，只是不会输出到后续指令的执行环境中。
- -p：列出所有的 shell 赋予程序的环境变量。

## enable

功能说明：启动或关闭 shell 内建指令。

语法：enable [-n][-all][内建指令]

使用说明：若要执行的文件名称与 shell 内建指令相同，可用 enable-n 来关闭 shell 内建指令。若不加-n 参数，enable 可重新启动关闭的指令。

参数：

- -n：关闭指定的 shell 内建指令。
- -all：显示 shell 所有关闭与启动的指令。

## dmesg

功能说明：显示开机信息。

语法：dmesg [-cn][-s<缓冲区大小>]

使用说明：kernel 会将开机信息存储在 ring buffer 中。您若是开机时来不及查看信息，可利用 dmesg 来查看。开机信息亦保存在/var/log 目录中，名称为 dmesg 的文件里。

参数：

- -c：显示信息后，清除 ring buffer 中的内容。
- -s<缓冲区大小>：预设置为 8196，刚好等于 ring buffer 的大小。
- -n：设置记录信息的层级。

## depmod

功能说明：分析可载入模块的相依性。

语法：depmod [-adeisvV][-m<文件>][--help][模块名称]

使用说明：depmod 可检测模块的相依性，供 modprobe 在安装模块时使用。

参数：

- -a 或--all：分析所有可用的模块。
- -d 或 debug：执行排错模式。
- -e：输出无法参照的符号。
- -i：不检查符号表的版本。
- -m<文件>或 system-map<文件>：使用指定的符号表文件。
- -s 或--system-log：在系统记录中记录错误。
- -v 或--verbose：执行时显示详细的信息。
- -V 或--version：显示版本信息。
- --help：显示帮助。

## su

功能说明：变更用户身份。

语法：su [-flmp][--help][--version][-][-c<指令>][-s<shell>][用户账号]

使用说明：su 可让用户暂时变更登录的身份。变更时需输入所要变更的用户账号与密码。

参数：

- -c<指令>或--command=<指令>：执行完指定的指令后，即恢复原来的身份。
- -f 或--fast：适用于 csh 与 tsch，使 shell 不用去读取启动文件。
- -.-l 或--login：改变身份时，也同时变更工作目录，以及 HOME、SHELL、USER、LOGNAME。此外，也会变更 PATH 变量。
- -m,-p 或--preserve-environment：变更身份时，不要变更环境变量。
- -s<shell>或--shell=<shell>：指定要执行的 shell。
- --help：显示帮助。
- --version：显示版本信息。
- 用户账号：指定要变更的用户。若不指定此参数，则预设变更为 root。

## lastb

功能说明：列出登录系统失败的用户的相关信息。

语法：lastb [-adRx][-f<记录文件>][-n<显示列数>][账号名称...][终端机编号...]

使用说明：单独执行 lastb 指令，它会读取位于/var/log 目录下，名称为 btmp 的文件，并把该文件内容记录的登录失败的用户名单全部显示出来。

参数：

- -a：把从何处登录系统的主机名称或 IP 地址显示在最后一行。
- -d：将 IP 地址转换成主机名称。
- -f<记录文件>：指定记录文件。
- -n<显示列数>或-<显示列数>：设置列出名单的显示列数。
- -R：不显示登录系统的主机名称或 IP 地址。
- -x：显示系统关机、重新开机以及执行等级的改变等信息。

## logrotate

功能说明：管理记录文件。

语法：logrotate [-?dfv][-s<状态文件>][--usage][配置文件]

使用说明：使用 logrotate 指令，可让你轻松管理系统所产生的记录文件。它提供自动替换、压缩、删除和邮寄记录文件，每个记录文件都可被设置成每日、每周或每月处理，也能在文件太大时立即处理。您必须自行编辑，指定配置文件。预设的配置文件存放在/etc 目录下，文件名称为 logrotate.conf。

参数：

- -?或--help：在线帮助。
- -d 或--debug：详细显示指令的执行过程，便于排错或了解程序执行的情况。
- -f 或--force：强行启动记录文件维护操作，即使 logrotate 指令认为没有必要也这样。
- -s<状态文件>或--state=<状态文件>：使用指定的状态文件。
- -v 或--version：显示指令的执行过程。
- -usage：显示指令的基本用法。

## newgrp

功能说明：登录另一个群组。

语法：newgrp [群组名称]

使用说明：newgrp 指令类似 login 指令，它是以相同的账号，另一个群组名称，再次登录系统。欲使用 newgrp 指令切换群组，您必须是该群组的用户，否则将无法登录指定的群组。单一用户要同时隶属多个群组，需利用交替用户的设置。若不指定群组名称，则 newgrp 指令会登录该用户名称的预设群组。

## procinfo

功能说明：显示系统状态。

语法：procinfo [-abdDfhimsSv][-F<输出文件>][-n<间隔秒数>]

使用说明：procinfo 指令从/proc 目录里读取相关数据，将数据妥善整理过后输出到标准输出设备。

参数：

- -a：显示所有信息。
- -b：显示磁盘设备的区块数目，而非存取数目。
- -d：显示系统信息每秒间的变化差额，而非总和的数值。本参数必须配合“-f”参数使用。
- -D：此参数效果和指定“-d”参数类似，但内存和交换文件的信息为总和数值。
- -f：进入全画面的互动式操作界面。
- -F<输出文件>：把信息状态输出到文件保存起来，而非预设的标准输出设备。
- -h：在线帮助。
- -i：显示完整的 IRP 列表。
- -m：显示系统模块和外围设备等相关信息。
- -n：间隔秒数>：设置全画面互动模式的信息更新速度，单位以秒计算。
- -s：显示系统的内存、磁盘空间、IRP 和 DMA 等信息，此为预设值。
- -S：搭配参数“-d”或“-D”使用时，每秒都会更新信息，不论是否使用参数“-n”。
- -v：显示版本信息。

## sudo

功能说明：以其他身份来执行指令。

语法：sudo [-bhHpV][-s<shell>][-u<用户>][指令]  或：sudo [-klv]

使用说明：sudo 可让用户以其他的身份来执行指定的指令，预设的身份为 root。在/etc/sudoers 中设置了可执行 sudo 指令的用户。若其未经授权的用户企图使用 sudo，则会发出警告邮件给管理员。用户使用 sudo 时，必须先输入密码，之后有 5 分钟的有效期限，超过期限则必须重新输入密码。

参数：

- -b：在后台执行指令。
- -h：显示帮助。
- -H：将 HOME 环境变量设为新身份的 HOME 环境变量。
- -k：结束密码的有效期限，也就是下次再执行 sudo 时便需要输入密码。
- -l：列出目前用户可执行与无法执行的指令。
- -p：改变询问密码的提示符号。
- -s<shell>：执行指定的 shell。
- -u<用户>：以指定的用户作为新的身份。若不加上此参数，则预设以 root 作为新的身份。
- -v：延长密码有效期限 5 分钟。
- -V：显示版本信息。

## suspend

功能说明：暂停执行 shell。

语法：suspend [-f]

使用说明：suspend 为 shell 内建指令，可暂停目前正在执行的 shell。若要恢复，则必须使用

SIGCONT 信息。

参数：

- -f：若目前执行的 shell 为登录的 shell，则 suspend 预设无法暂停此 shell。若要强迫暂停登录的 shell，则必须使用-f 参数。

## swatch

功能说明：系统监控程序。

语法：swatch [-A<分隔字符>][-c<设置文件>][-f<记录文件>][-I<分隔字符>][-P<分隔字符>][-r<时间>][-t<记录文件>]

使用说明：swatch 可用来监控系统记录文件，并在发现特定的事件时，执行指定的动作。swatch 所监控的事件以及对应事件的动作都存放在 swatch 的配置文件中。预设的配置文件为用户根目录下的.swatchrc。然而在 Red Hat Linux 的预设用户根目录下并没有.swatchrc 配置文件，您可将/usr/doc/swatch-2.2/config_files/swatchrc.personal 文件复制到用户根目录下的.swatchrc，然后修改.swatchrc 所要监控的事件及执行的动作。

参数：

- -A<分隔字符>：指定配置文件中，动作的分隔字符，预设为逗号。
- -c<设置文件>：指定配置文件，而不使用预设的配置文件。
- -f<记录文件>：检查指定的记录文件，检查完毕后不会继续监控该记录文件。
- -I<分隔字符>：指定输入记录的分隔字符，预设为换行字符。
- -P<分隔字符>：指定配置文件中，事件的分隔字符，预设为逗号。
- -r<时间>：在指定的时间重新启动。
- -t<记录文件>：检查指定的记录文件，并且会监控加入记录文件中的后继记录。

## tload

功能说明：显示系统负载状况。

语法：tload [-V][-d<间隔秒数>][-s<刻度大小>][终端机编号]

使用说明：tload 指令使用 ASCII 字符简单地以文字模式显示系统的负载状态。假设不给予终端机编号，则会在执行 tload 指令的终端机显示负载情形。

参数：

- -d<间隔秒数>：设置 tload 检测系统负载的间隔时间，单位以秒计算。
- -s<刻度大小>：设置图表的垂直刻度大小，单位以列计算。
- -V：显示版本信息。

## uname

功能说明：显示系统信息。

语法：uname [-amnrsv][--help][--version]

使用说明：uname 可显示电脑以及操作系统的相关信息。

参数：

- -a 或--all：显示全部的信息。

- -m 或--machine：显示电脑类型。
- -n 或-nodename：显示在网络上的主机名称。
- -r 或--release：显示操作系统的发行编号。
- -s 或--sysname：显示操作系统名称。
- -v：显示操作系统的版本。
- --help：显示帮助。
- --version：显示版本信息。

## userconf

功能说明：用户账号设置程序。

语法：userconf [--addgroup<群组>][--adduser<用户 ID><群组><用户名称><shell>][--delgroup<群组>][--deluser<用户 ID>][--help]

使用说明：userconf 实际上为 linuxconf 的符号连接，提供图形界面的操作方式，供管理员建立与管理各类账号。若不加任何参数，即进入图形界面。

参数：

- --addgroup<群组>：新增群组。
- --adduser<用户 ID><群组><用户名称><shell>：新增用户账号。
- --delgroup<群组>：删除群组。
- --deluser<用户 ID>：删除用户账号。
- --help：显示帮助。

## vlock

功能说明：锁住虚拟终端。

语法：vlock [-achv]

使用说明：执行 vlock 指令可锁住虚拟终端，避免他人使用。

参数：

- -a 或--all：锁住所有的终端阶段作业，如果您在全屏幕的终端中使用本参数，则会将用键盘切换终端机的功能一并关闭。
- -c 或--current：锁住目前的终端阶段作业，此为预设值。
- -h 或--help：在线帮助。
- -v 或--version：显示版本信息。

## whoami

功能说明：显示用户名称。

语法：whoami [--help][--version]

使用说明：显示自身的用户名称，本指令相当于执行“id-un”指令。

参数：

- --help：在线帮助。
- --version：显示版本信息。

### whois

功能说明：查找并显示用户信息。

语法：whois [账号名称]

使用说明：whois 指令会去查找并显示指定账号的用户相关信息，因为它是到 Network Solutions 的 WHOIS 数据库去查找，所以该账号名称必须在上面注册方能寻获，且名称没有大小写的区别。

## A.3 系统设置命令

### alias

功能说明：设置指令的别名。

语法：alias[别名]=[指令名称]

使用说明：用户可利用 alias 自定指令的别名。若仅输入 alias，则可列出目前所有的别名设置。alias 的效力仅及于该次登录的操作。若要每次登录时即自动设好别名，可在.profile 或.cshrc 中设定指令的别名。

参数：若不加任何参数，则列出目前所有的别名设置。

### unset

功能说明：删除变量或函数。

语法：unset [-fv][变量或函数名称]

使用说明：unset 为 shell 内建指令，可删除变量或函数。

参数：

- -f：仅删除函数。
- -v：仅删除变量。

### unalias

功能说明：删除别名。

语法：unalias [-a][别名]

使用说明：unalias 为 shell 内建指令，可删除别名设置。

参数：

- -a：删除全部的别名。

### ulimit

功能说明：控制 shell 程序的资源。

语法：ulimit [-aHS][-c<core 文件上限>][-d<数据块区大小>][-f<文件大小>][-m<内存大小>][-n<文件数目>][-p<缓冲区大小>][-s<堆叠大小>][-t<CPU 时间>][-u<程序数目>][-v<虚拟内存大小>]

使用说明：ulimit 为 shell 内建指令，可用来控制 shell 执行程序的资源。

参数：

- -a：显示目前资源限制的设定。
- -c<core 文件上限>：设定 core 文件的最大值，单位为区块。
- -d<数据节区大小>：程序数据块区的最大值，单位为 KB。
- -f<文件大小>：shell 所能建立的最大文件，单位为区块。
- -H：设定资源的硬性限制，也就是管理员所设置的限制。
- -m<内存大小>：指定可使用内存的上限，单位为 KB。
- -n<文件数目>：指定同一时间最多可开启的文件数。
- -p<缓冲区大小>：指定管道缓冲区的大小，单位为 512KB。
- -s<堆叠大小>：指定堆叠的上限，单位为 KB。
- -S：设定资源的弹性限制。
- -t<CPU 时间>：指定 CPU 使用时间的上限，单位为秒。
- -u<程序数目>：用户最多可开启的程序数目。
- -v<虚拟内存大小>：指定可使用的虚拟内存上限，单位为 KB。

## set

功能说明：设置 shell。

语法：set [+-abCdefhHklmnpPtuvx]

使用说明：set 指令能设置所使用 shell 的执行方式，可依照不同的需求来进行设置。

参数：

- -a：标示已修改的变量，以供输出至环境变量。
- -b：使被中止的后台程序立刻回报执行状态。
- -C：转向所产生的文件，无法覆盖已存在的文件。
- -d：shell 预设会用杂凑表记忆使用过的指令，以加速指令的执行。使用-d 参数可取消。
- -e：若指令传回值不等于 0，则立即退出 shell。
- -f：取消使用通配符。
- -h：自动记录函数的所在位置。
- -H：shell：可利用“!”加<指令编号>的方式来执行 history 中记录的指令。
- -k：指令所给的参数都会被视为此指令的环境变量。
- -l：记录 for 循环的变量名称。
- -m：使用监视模式。
- -n：只读取指令，而不实际执行。
- -p：启动优先顺序模式。
- -P：启动-P 参数后，执行指令时，会以实际的文件或目录来取代符号连接。
- -t：执行完随后的指令即退出 shell。
- -u：当执行时使用到未定义过的变量，则显示错误信息。
- -v：显示 shell 所读取的输入值。
- -x：执行指令后，会先显示该指令及其下的参数。
- +<参数>：取消某个 set 曾启动的参数。

## rmmod

功能说明：删除模块。

语法：rmmod [-as][模块名称...]

使用说明：执行 rmmod 指令，可删除不需要的模块。Linux 操作系统的核心具有模块化的特性，因此在编译核心时，必须把全部的功能都放入核心。你可以将这些功能编译成一个个单独的模块，待有需要时再分别载入它们。

参数：

- -a：删除所有目前不需要的模块。
- -s：把信息输出至 syslog 常驻服务，而非终端机界面。

## resize

功能说明：设置终端机视窗的大小。

语法：resize [-cu][-s<列数><行数>]

使用说明：执行 resize 指令可设置虚拟终端机的视窗大小。

参数：

- -c：就算用户环境并非 C Shell，也用 C Shell 指令改变视窗大小。
- -s<列数><行数>：设置终端机视窗的垂直高度和水平宽度。
- -u：就算用户环境并非 Bourne Shell，也用 Bourne Shell 指令改变视窗大小。

## sndconfig

功能说明：设置声卡。

语法：sndconfig [--help][--noautoconfig][--noprobe]

使用说明：sndconfig 为声卡设置程序，支持 PnP 设置，可自动检测并设置 PnP 声卡。

参数：

- --help：显示帮助。
- --noautoconfig：不自动设置 PnP 的声卡。
- --noprobe：不自动检测 PnP 声卡。

## setup

功能说明：设置公用程序。

语法：setup

使用说明：setup 是一个设置公用程序、提供图形界面的操作方式。在 setup 中可设置以下 7 类选项。

- 登录认证方式。
- 键盘组态设置。
- 鼠标组态设置。
- 开机时所要启动的系统服务。
- 声卡组态设置。
- 时区设置。

- X Windows 组态设置。

## setenv

功能说明：查询或显示环境变量。

语法：setenv [变量名称][变量值]

使用说明：setenv 为 tsch 中查询或设置环境变量的指令。

## setconsole

功能说明：设置系统终端。

语法：setconsole [video][serial][ttya][ttyb]

使用说明：setconsole 可用来指定系统终端。

参数：

- serial：使用 PROM 终端。
- ttya,cua0 或 ttyS0：使用第 1 个串口设备作为终端。
- ttyb,cua1 或 ttyS1：使用第 2 个串口设备作为终端。
- video：使用主机上的现卡作为终端。

## reset

语法：tset[-IQqrs][-][-e ch][-I ch][-k ch][-m mapping][terminal]

使用说明：reset 其实和 tset 是同一个命令，它的用途是设定终端机的状态。通常这个命令会自动地从环境函数、命令列或是其他的组态档决定目前终端机的型态。如果指定型态是：'?'：的话，这个程序会要求使用者输入终端机的型别。由于这个程序会将终端机设回原始的状态，除了在 login 时使用外，当系统终端机因为程序不正常执行而进入一些奇怪的状态时，你也可以用它来重设终端机；例如不小心把二进位挡用 cat 指令进到终端机，常会有终端机不再回应键盘输入，或是回应一些奇怪字元的问题，此时就可以用 reset 将终端机恢复至原始状态。

参数

- -p：将终端机类别显示在屏幕上，但不做设定的动作。这个命令可以用来取得目前终端机的类别。
- -e ch：将 erase 字元设成 ch。
- -i ch：将中断字元设成 ch。
- -k ch：将删除一行的字元设成 ch。
- -I：不要做设定的动作，如果没有使用选项-Q 的话，erase、中断及删除字元的目前值依然会送到屏幕上。
- -Q：不要显示 erase 中断及删除字元的值到屏幕上。
- -r：将终端机类别印在屏幕上。
- -s：将设定 TERM 用的命令用字串的形式送到终端机中，通常在.login 或 profile 中用。

## pwunconv

功能说明：关闭用户的投影密码。

语法：pwunconv

使用说明：执行 pwunconv 指令可以关闭用户投影密码，它会把密码从 shadow 文件内，重回存到 passwd 文件里。

## pwconv

功能说明：开启用户的投影密码。

语法：pwconv

使用说明：Linux 系统里的用户和群组密码，分别存放在名称为 passwd 和 group 的文件中，这两个文件位于/etc 目录下。因系统运作所需，任何人都得以读取它们，造成安全上的破绽。投影密码将文件内的密码改存在/etc 目录下的 shadow 和 gshadow 文件内，只允许系统管理员读取，同时把原密码置换为“x”字符，有效地强化了系统的安全性。

## passwd

功能说明：设置密码。

语法：passwd [-dklS][-u<-f>][用户名称]

使用说明：passwd 指令让用户可以更改自己的密码，而系统管理者则能用它管理系统用户的密码。只有管理者可以指定用户名称，一般用户只能变更自己的密码。

参数：

- -d：删除密码。本参数仅有系统管理者才能使用。
- -f：强制执行。
- -k：设置只有在密码过期失效后，方能更新。
- -l：锁住密码。
- -S：列出密码的相关信息。本参数仅有系统管理者才能使用。
- -u：解开已上锁的账号。

## ntsysv

功能说明：设置系统的各种服务。

语法：ntsysv [--back][--level<等级代号>]

使用说明：这是 Red Hat 公司遵循 GPL 规则所开发的程序，它具有互动式操作界面，您可以轻易地利用方向键和空格键等开启、关闭操作系统在每个执行等级中所要执行的系统服务。

参数：

- --back：在互动式界面里，显示 Back 按钮，而非 Cancel 按钮。
- --level<等级代号>：在指定的执行等级中，决定要开启或关闭哪些系统服务。

## modprobe

功能说明：自动处理可载入模块。

语法：modprobe [-acdlrtvV][--help][模块文件][符号名称=符号值]

使用说明：modprobe 可载入指定的个别模块，或是载入一组相依的模块。modprobe 会根据 depmod 所产生的相依关系，决定要载入哪些模块。若在载入过程中发生错误，modprobe 会卸载整组的模块。

参数：

- -a 或--all：载入全部的模块。
- -c 或--show-conf：显示所有模块的设置信息。
- -d 或--debug：使用排错模式。
- -l 或--list：显示可用的模块。
- -r 或--remove：模块闲置不用时即自动卸载。
- -t 或--type：指定模块类型。
- -v 或--verbose：执行时显示详细的信息。
- -V 或--version：显示版本信息。
- -help：显示帮助。

**modinfo**

功能说明：显示 kernel 模块的信息。

语法：modinfo [-adhpV][模块文件]

使用说明：modinfo 会显示 kernel 模块的对象文件，以显示该模块的相关信息。

参数：

- -a 或--author：显示模块开发人员。
- -d 或--description：显示模块的说明。
- -h 或--help：显示 modinfo 的参数使用方法。
- -p 或--parameters：显示模块所支持的参数。
- -V 或--version：显示版本信息。

**mkkickstart**

功能说明：建立安装的组态文件。

语法：mkkickstart [--bootp][--dhcp][--nonet][--nox][--version][--nfs<远端电脑:路径>]

使用说明：mkkickstart 可根据目前系统的设置来建立组态文件，供其他电脑在安装时使用。组态文件的内容包括使用语言、网络环境、系统磁盘状态，以及 X Windows 的设置等信息。

参数：

- --bootp：安装与开机时使用 BOOTP。
- --dhcp：安装与开机时使用 DHCP。
- --nfs<远端电脑:路径>：使用指定的网络路径安装。
- --nonet：不要进行网络设置，即假设在没有网络环境的状态下。
- --nox：不要进行 X Windows 的环境设置。
- --version：显示版本信息。

**chroot**

功能说明：改变根目录。

语法：chroot [--help][--version][目的目录][执行指令...]

使用说明：把根目录换成指定的目的目录。

参数：

- --help：在线帮助。

- --version：显示版本信息。

### chkconfig

功能说明：检查，设置系统的各种服务。

语法：chkconfig [--add][--del][--list][系统服务]或：chkconfig [--level<等级代号>][系统服务][on/off/reset]

使用说明：这是 Red：Hat 公司遵循 GPL 规则所开发的程序，它可查询操作系统在每一个执行等级中会执行哪些系统服务，其中包括各类常驻服务。

参数：

- --add：增加所指定的系统服务，让 chkconfig 指令得以管理它，并同时在系统启动的叙述文件内增加相关数据。
- --del：删除所指定的系统服务，不再由 chkconfig 指令管理，并同时在系统启动的叙述文件内删除相关数据。
- --level<等级代号>：指定读系统服务要在哪一个执行等级中开启或关闭。

## A.4 磁盘管理及维护命令

### cd

功能说明：切换目录。

语法：cd [目的目录]

使用说明：cd 指令可让用户在不同的目录间切换，但该用户必须拥有足够的权限进入目的目录。

### df

功能说明：显示磁盘的相关信息。

语法：df [-ahHiklmPT][--block-size=<区块大小>][-t<文件系统类型>][-x<文件系统类型>][--help][--no-sync][--sync][--version][文件或设备]

使用说明：df 可显示磁盘的文件系统与使用情形。

参数：

- -a 或--all：包含全部的文件系统。
- --block-size=<区块大小>：以指定的区块大小来显示区块数目。
- -h 或--human-readable：以可读性较高的方式来显示信息。
- -H 或--si：与-h 参数相同，但在计算时是以 1000 Bytes 为换算单位而非 1024 Bytes。
- -i 或--inodes：显示 inode 的信息。
- -k 或--kilobytes：指定区块大小为 1024 字节。
- -l 或--local：仅显示本地端的文件系统。
- -m 或--megabytes：指定区块大小为 1 048 576 字节。
- --no-sync：在取得磁盘使用信息前不要执行 sync 指令，此为预设值。

- -P 或--portability：使用 POSIX 输出格式。
- --sync：在取得磁盘使用信息前，先执行 sync 指令。
- -t<文件系统类型>或--type=<文件系统类型>：仅显示指定文件系统类型的磁盘信息。
- -T 或--print-type：显示文件系统的类型。
- -x<文件系统类型>或--exclude-type=<文件系统类型>：不要显示指定文件系统类型的磁盘信息。
- --help：显示帮助。
- --version：显示版本信息。
- 文件或设备：指定磁盘设备。

## dirs

功能说明：显示目录记录。

语法：dirs [+/-n-l]

使用说明：显示目录堆叠中的记录。

参数：

- +n：显示从左边算起第 n 笔的目录。
- -n：显示从右边算起第 n 笔的目录。
- -l：显示目录完整的记录。

## du

功能说明：显示目录或文件的大小。

语法：du [-abcDhHklmsSx][-L<符号连接>][-X<文件>][--block-size][--exclude=<目录或文件>][--max-depth=<目录层数>][--help][--version][目录或文件]

使用说明：du 会显示指定的目录或文件所占用的磁盘空间。

参数：

- -a 或-all：显示目录中个别文件的大小。
- -b 或-bytes：显示目录或文件大小时，以 byte 为单位。
- -c 或--total：除了显示个别目录或文件的大小外，同时也显示所有目录或文件的总和。
- -D 或--dereference-args：显示指定符号连接的源文件大小。
- -h 或--human-readable：以 K、M、G 为单位，提高信息的可读性。
- -H 或--si：与-h 参数相同，但是 K、M、G 是以 1000 为换算单位。
- -k 或--kilobytes：以 1024 bytes 为单位。
- -l 或--count-links：重复计算硬件连接的文件。
- -L<符号连接>或--dereference<符号连接>：显示选项中所指定符号连接的源文件大小。
- -m 或--megabytes：以 1MB 为单位。
- -s 或--summarize：仅显示总计。
- -S 或--separate-dirs：显示个别目录的大小时，并不含其子目录的大小。
- -x 或--one-file-xystem：以一开始处理时的文件系统为准，若遇上其他不同的文件系统目录则略过。
- -X<文件>或--exclude-from=<文件>：在文件指定目录或文件。

- --exclude=<目录或文件>：略过指定的目录或文件。
- --max-depth=<目录层数>：超过指定层数的目录后，予以忽略。
- --help：显示帮助。
- --version：显示版本信息。

## edquota

功能说明：编辑用户或群组的 quota。

语法：edquota [-p<源用户名称>][-ug][用户或群组名称...] 或：edquota [-ug]-t

使用说明：edquota 预设会使用 vi 来编辑使用者或群组的 quota 设置。

参数：

- -u：设置用户的 quota，这是预设的参数。
- -g：设置群组的 quota。
- -p<源用户名称>：将源用户的 quota 设置套用至其他用户或群组。
- -t：设置宽限期限。

## eject

功能说明：退出抽取式设备。

语法：eject [-dfhnqrstv][-a<开关>][-c<光驱编号>][设备]

使用说明：若设备已挂入，则 eject 会先将该设备卸除再退出。

参数：

- 设备：设备可以是驱动程序名称，也可以是挂入点。
- -a<开关>或--auto<开关>：控制设备的自动退出功能。
- -c<光驱编号>或--changerslut<光驱编号>：选择光驱柜中的光驱。
- -d 或--default：显示预设的设备，而不是实际执行动作。
- -f 或--floppy：退出抽取式磁盘。
- -h 或--help：显示帮助。
- -n 或--noop：显示指定的设备。
- -q 或--tape：退出磁带。
- -r 或--cdrom：退出光盘。
- -s 或--scsi：以 SCSI 指令来退出设备。
- -t 或--trayclose：关闭光盘的托盘。
- -v 或--verbose：执行时，显示详细的说明。

## lndir

功能说明：连接目录内容。

语法：lndir [-ignorelinks][-silent][源目录][目的目录]

使用说明：执行 lndir 指令，可一口气把源目录底下的文件和子目录统统建立起相互对应的符号连接。

参数：

- -ignorelinks：直接建立符号连接。

- -silent：不显示指令的执行过程。

## ls

功能说明：列出目录内容。

语法：ls [-1aAbBcCdDfFgGhHiklLmnNopqQrRsStuUvxX][-I<范本样式>][-T<跳格字数>][-w <每列字符数>][--block-size=<区块大小>][--color=<使用时机>][--format=<列表格式>][--full-time][--help][--indicator-style=<标注样式>][--quoting-style=<引号样式>][--show-control-chars][--sort=<排序方式>][--time=<时间戳记>][--version][文件或目录...]

使用说明：执行 ls 指令可列出目录的内容，包括文件和子目录的名称。

参数：

- -1：每列仅显示一个文件或目录名称。
- -a 或--all：显示所有文件和目录。
- -A 或--almost-all：显示所有文件和目录，但不显示现行目录和上层目录。
- -b 或--escape：显示脱离字符。
- -B 或--ignore-backups：忽略备份文件和目录。
- -c：以更改时间排序，显示文件和目录。
- -C：以由上至下，从左到右的直行方式显示文件和目录名称。
- -d 或--directory：显示目录名称而非其内容。
- -D 或--dired：用 Emacs 的模式产生文件和目录列表。
- -f：此参数的效果和同时指定“aU”参数相同，并关闭“lst”参数的效果。
- -F 或--classify：在执行文件、目录、Socket、符号连接、管道名称后面，各自加上“*”、“/”、“=”、“@”、“|”号。
- -g：此参数将忽略不予处理。
- -G 或--no-group：不显示群组名称。
- -h 或--human-readable：用“K”、“M”、“G”来显示文件和目录的大小。
- -H 或--si：此参数的效果和指定“-h”参数类似，但计算单位是 1000Bytes 而非 1024Bytes。
- -i 或--inode：显示文件和目录的 inode 编号。
- -I<范本样式>或--ignore=<范本样式>：不显示符合范本样式的文件或目录名称。
- -k 或--kilobytes：此参数的效果和指定“block-size=1024”参数相同。
- -l：使用详细格式列表。
- -L 或--dereference：如遇到性质为符号连接的文件或目录，直接列出该连接所指向的原始文件或目录。
- -m：用“,”号分隔每个文件和目录的名称。
- -n 或--numeric-uid-gid：以用户识别码和群组识别码替代其名称。
- -N 或--literal：直接列出文件和目录名称，包括控制字符。
- -o：此参数的效果和指定“-l”：参数类似，但不列出群组名称或识别码。
- -p 或--file-type：此参数的效果和指定“-F”参数类似，但不会在执行文件名称后面加上“*”号。
- -q 或--hide-control-chars：用“?”号取代控制字符，列出文件和目录名称。

- -Q 或--quote-name：把文件和目录名称以“”号标示起来。
- -r 或--reverse：反向排序。
- -R 或--recursive：递归处理，将指定目录下的所有文件及子目录一并处理。
- -s 或--size：显示文件和目录的大小，以区块为单位。
- -S：用文件和目录的大小排序。
- -t：用文件和目录的更改时间排序。
- -T<跳格字符>或--tabsize=<跳格字数>：设置跳格字符所对应的空白字符数。
- -u：以最后存取时间排序，显示文件和目录。
- -U：列出文件和目录名称时不予排序。
- -v：文件和目录的名称列表以版本进行排序。
- -w<每列字符数>或--width=<每列字符数>：设置每列的最大字符数。
- -x：以从左到右，由上至下的横列方式显示文件和目录名称。
- -X：以文件和目录的最后一个扩展名排序。
- --block-size=<区块大小>：指定存放文件的区块大小。
- --format=<列表格式>：配置文件和目录的列表格式。
- --full-time：列出完整的日期与时间。
- --help：在线帮助。
- --indicator-style=<标注样式>：在文件和目录等名称后面加上标注，易于辨识该名称所属的类型。
- --quoting-syte=<引号样式>：把文件和目录名称以指定的引号样式标示起来。
- --show-control-chars：在文件和目录列表时，使用控制字符。
- --sort=<排序方式>：配置文件和目录列表的排序方式。
- --time=<时间戳记>：用指定的时间戳记取代更改时间。
- --version：显示版本信息。

## mkdir

功能说明：建立目录

语法：mkdir [-p] [--help] [verbose] [--version] [-m<目录属性>] [目录名称]

使用说明：mkdir 可建立目录并同时设置目录的权限。

参数：

- -m<目录属性>或--mode<目录属性>：建立目录时同时设置目录的权限。
- -p 或--parents：若所要建立目录的上层目录尚未建立，则会一并建立上层目录。
- --help：显示帮助。
- --verbose：执行时显示详细的信息。
- --version：显示版本信息。

## pwd

功能说明：显示工作目录。

语法：pwd [--help][--version]

使用说明：执行 pwd 指令可立刻得知您目前所在的工作目录的绝对路径名称。

参数：

- --help：在线帮助。
- --version：显示版本信息。

## quota

功能说明：显示磁盘已使用的空间与限制。

语法：quota [-quvV][用户名称...] 或：quota [-gqvV][群组名称...]

使用说明：执行 quota 指令，可查询磁盘空间的限制，并得知已使用了多少空间。

参数：

- -g：列出群组的磁盘空间限制。
- -q：简明列表，只列出超过限制的部分。
- -u：列出用户的磁盘空间限制。
- -v：显示该用户或群组，在所有挂入系统的存储设备的空间限制。
- -V：显示版本信息。

## quotacheck

功能说明：检查磁盘的使用空间与限制。

语法：quotacheck [-adgRuv][文件系统...]

使用说明：执行 quotacheck 指令，扫描挂入系统的分区，并在各分区的文件系统根目录下产生 quota.user 和 quota.group 文件，设置用户和群组的磁盘空间限制。

参数：

- -a：扫描在/etc/fstab 文件里，有加入 quota 设置的分区。
- -d：详细显示指令的执行过程，便于排错或了解程序执行的情形。
- -g：扫描磁盘空间时，计算每个群组识别码所占用的目录和文件数目。
- -R：排除根目录所在的分区。
- -u：扫描磁盘空间时，计算每个用户识别码所占用的目录和文件数目。
- -v：显示指令的执行过程。

## tree

功能说明：以树状图列出目录的内容。

语法：tree [-aACdDfFgilnNpqstux][-I<范本样式>][-P<范本样式>][目录...]

使用说明：执行 tree 指令，它会列出指定目录下的所有文件，包括子目录里的文件。

参数：

- -a：显示所有文件和目录。
- -A：使用 ASNI 绘图字符显示树状图而非以 ASCII 字符组合。
- -C：在文件和目录清单加上色彩，便于区分各种类型。
- -d：显示目录名称而非内容。
- -D：列出文件或目录的更改时间。
- -f：在每个文件或目录之前，显示完整的相对路径名称。
- -F：在执行文件、目录、Socket、符号连接、管道名称后面、各自加上“*”、“/”、“=”、

"@"、"|"号。

- -g：列出文件或目录的所属群组名称，没有对应的名称时，则显示群组识别码。
- -i：不以阶梯状列出文件或目录名称。
- -I<范本样式>：不显示符合范本样式的文件或目录名称。
- -l：如遇到性质为符号连接的目录，直接列出该连接所指向的原始目录。
- -n：不在文件和目录清单加上色彩。
- -N：直接列出文件和目录名称，包括控制字符。
- -p：列出权限标示。
- -P<范本样式>：只显示符合范本样式的文件或目录名称。
- -q：用"?"号取代控制字符，列出文件和目录名称。
- -s：列出文件或目录大小。
- -t：用文件和目录的更改时间排序。
- -u：列出文件或目录的拥有者名称，没有对应的名称时，则显示用户识别码。
- -x：将范围局限在现行的文件系统中，若指定目录下的某些子目录，存放于另一个文件系统上，则将该子目录予以排除在寻找范围外。

## umount

功能说明：卸除文件系统。

语法：umount [-ahnrvV][-t<文件系统类型>][文件系统]

使用说明：umount 可卸除目前挂在 Linux 目录中的文件系统。

参数：

- -a：卸载/etc/mtab 中记录的所有文件系统。
- -h：显示帮助。
- -n：卸载时不要将信息存入/etc/mtab 文件中。
- -r：若无法成功卸载，则尝试以只读方式重新挂入文件系统。
- -t<文件系统类型>：仅卸载选项中所指定的文件系统。
- -v：执行时显示详细的信息。
- -V：显示版本信息。
- 文件系统：除了直接指定文件系统外，也可以用设备名称或挂入点来表示文件系统。

## sync

功能说明：将内存缓冲区内的数据写入磁盘。

语法：sync [--help][--version]

使用说明：在 Linux 系统中，当数据需要存入磁盘时，通常会先放到缓冲区内，等到适当的时刻再写入磁盘，这样可提高系统的执行效率。

参数：

- --help：显示帮助。
- --version：显示版本信息。

## symlinks

功能说明：维护符号连接的工具程序。

语法：symlinks [-cdrstv][目录]

使用说明：symlinks可检查目录中的符号连接，并显示符号连接类型。以下为symlinks可判断的符号连接类型。

- absolute：符号连接使用了绝对路径。
- dangling：原始文件已经不存在。
- lengthy：符号连接的路径中包含了多余的“../”。
- messy：符号连接的路径中包含了多余的“/”。
- other_fs：原始文件位于其他文件系统中。
- relative：符号连接使用了相对路径。

参数：

- -c：将使用绝对路径的符号连接转换为相对路径。
- -d：移除dangling类型的符号连接。
- -r：检查目录下所有子目录中的符号连接。
- -s：检查lengthy类型的符号连接。
- -t：与-c一并使用时，会显示如何将绝对路径的符号连接转换为相对路径，但不会实际转换。
- -v：显示所有类型的符号连接。

## swapon

功能说明：启动系统交换区（swap area）。

语法：swapon [-ahsV][-p<优先顺序>][设备]

使用说明：Linux系统的内存管理必须使用交换区来建立虚拟内存。

参数：

- -a：将/etc/fstab文件中所有设置为swap的设备启动为交换区。
- -h：显示帮助。
- -p<优先顺序>：指定交换区的优先顺序。
- -s：显示交换区的使用状况。
- -V：显示版本信息。

## swapoff

功能说明：关闭系统交换区（swap area）。

语法：swapoff [设备]

使用说明：swapoff实际上为swapon的符号连接，可用来关闭系统的交换区。

## mkswap

功能说明：设置交换区（swap area）。

语法：mkswap [-cf][-v0][-v1][设备名称或文件][交换区大小]

使用说明：mkswap 可将磁盘分区或文件设为 Linux 的交换区。

参数：

- -c：建立交换区前，先检查是否有损坏的区块。
- -f：在 SPARC 电脑上建立交换区时，要加上此参数。
- -v0：建立旧式交换区，此为预设值。
- -v1：建立新式交换区。
- 交换区大小：指定交换区的大小，单位为 1024 字节。

## mkinitrd

功能说明：建立要载入 ramdisk 的映像文件。

语法：mkinitrd [-fv][--omit-scsi-modules][--version][--preload=<模块名称>][--with=<模块名称>][映像文件]

使用说明：mkinitrd 可建立映像文件，以供 Linux 开机时载入 ramdisk。

参数：

- -f：若指定的映像文件名称与现有文件重复，则覆盖现有的文件。
- -v：执行时显示详细的信息。
- --omit-scsi-modules：不要载入 SCSI 模块。
- --preload=<模块名称>：指定要载入的模块。
- --with=<模块名称>：指定要载入的模块。
- --version：显示版本信息。

## mkfs

功能说明：建立各种文件系统。

语法：mkfs [-vV][fs][-f<文件系统类型>][设备名称][区块数]

使用说明：mkfs 本身并不执行建立文件系统的工作，而是去调用相关的程序来执行。

参数：

- fs：指定建立文件系统时的参数。
- -f<文件系统类型>：指定要建立何种文件系统。
- -v：显示版本信息与详细的使用方法。
- -V：显示简要的使用方法。

## mke2fs

功能说明：建立 ext2 文件系统。

语法：mke2fs [-cFMqrSvV][-b<区块大小>][-f<不连续区段大小>][-i<字节>][-N<inode 数>][-l<文件>][-L<标签>][-m<百分比值>][-R=<区块数>][设备名称][区块数]

使用说明：mke2fs 可建立 Linux 的 ext2 文件系统。

参数：

- -b<区块大小>：指定区块大小，单位为字节。
- -c：检查是否有损坏的区块。

- -f<不连续区段大小>：指定不连续区段的大小，单位为字节。
- -F：不管指定的设备为何，强制执行 mke2fs。
- -i<字节>：指定“字节/inode”的比例。
- -N<inode 数>：指定要建立的 inode 数目。
- -l<文件>：从指定的文件中，读取文件被损坏区块的信息。
- -L<标签>：设置文件系统的标签名称。
- -m<百分比值>：指定给管理员保留区块的比例，预设为 5%。
- -M：记录最后一次挂入的目录。
- -q：执行时不显示任何信息。
- -r：指定要建立的 ext2 文件系统版本。
- -R=<区块数>：设置磁盘阵列参数。
- -S：仅写入 superblock 与 group descriptors，而不更改 inode able、inode bitmap 以及 block bitmap。
- -v：执行时显示详细信息。
- -V：显示版本信息。

### mkbootdisk

功能说明：建立目前系统的启动盘。

语法：mkbootdisk [--noprompt][--verbose][--version][--device<设备>][--mkinitrdargs<参数>][kernel：版本]

使用说明：mkbootdisk 可建立目前系统的启动盘。

参数：

- --device<设备>：指定设备。
- --mkinitrdargs<参数>：设置 mkinitrd 的参数。
- --noprompt：不会提示用户插入磁盘。
- --verbose：执行时显示详细的信息。
- --version：显示版本信息。

### quotaoff

功能说明：关闭磁盘空间限制。

语法：quotaoff [-aguv][文件系统...]

使用说明：执行 quotaoff 指令可关闭用户和群组的磁盘空间限制。

参数：

- -a：关闭在/etc/fstab 文件里加入 quota 设置的分区的空间限制。
- -g：关闭群组的磁盘空间限制。
- -u：关闭用户的磁盘空间限制。
- -v：显示指令的执行过程。

### quotaon

功能说明：开启磁盘空间限制。

语法：quotaon [-aguv][文件系统...]

使用说明：执行 quotaon 指令可开启用户和群组的磁盘空间限制，各分区的文件系统根目录必须有 quota.user 和 quota.group 配置文件。

参数：

- -a：开启在/ect/fstab 文件里加入 quota 设置的分区的空间限制。
- -g：开启群组的磁盘空间限制。
- -u：开启用户的磁盘空间限制。
- -v：显示指令的执行过程。

## repquota

功能说明：检查磁盘空间限制的状态。

语法：repquota [-aguv][文件系统...]

使用说明：执行 repquota 指令，可报告磁盘空间限制的状况，清楚地得知每位用户或每个群组已使用多少空间。

参数：

- -a：列出在/etc/fstab 文件里，加入 quota 设置的分区的使用状况，包括用户和群组。
- -g：列出所有群组的磁盘空间限制。
- -u：列出所有用户的磁盘空间限制。
- -v：显示该用户或群组的所有空间限制。

## rmdir

功能说明：删除目录。

语法：rmdir [-p][--help][--ignore-fail-on-non-empty][--verbose][--version][目录...]

使用说明：当有空目录要删除时，可使用 rmdir 指令。

参数：

- -p 或--parents：删除指定目录后，若该目录的上层目录已变成空目录，则将其一并删除。
- --help：在线帮助。
- --ignore-fail-on-non-empty：忽略非空目录的错误信息。
- --verbose：显示指令的执行过程。
- --version：显示版本信息。

## stat

功能说明：显示 inode 内容。

语法：stat [文件或目录]

使用说明：stat 以文字的格式来显示 inode 的内容。

## hdparm

功能说明：显示与设定硬盘的参数。

语法：hdparm [-CfghiIqtTvyYZ][-a<快取分区>][-A<0 或 1>][-c<I/O 模式>][-d<0 或 1>][-k<0 或 1>][-K<0 或 1>][-m<分区数>][-n<0 或 1>][-p<PIO 模式>][-P<分区数>][-r<0 或 1>][-S<时

间>][-u<0 或 1>][-W<0 或 1>][-X<传输模式>][设备]

使用说明：hdparm 可检测、显示与设定 IDE 或 SCSI 硬盘的参数。

参数：

- -a<块区分区>：设定读取文件时，预先存入块区的分区数，若不加上“块区分区”选项，则显示目前的设定。
- -A<0 或 1>：启动或关闭读取文件时的块区功能。
- -c<I/O 模式>：设定 IDE32 位 I/O 模式。
- -C：检测 IDE 硬盘的电源管理模式。
- -d<0 或 1>：设定磁盘的 DMA 模式。
- -f：将内存缓冲区的数据写入硬盘，并清除缓冲区。
- -g：显示硬盘的磁轨、磁头、磁区等参数。
- -h：显示帮助。
- -i：显示硬盘的硬件规格信息，这些信息是在开机时由硬盘本身所提供。
- -I：直接读取硬盘所提供的硬件规格信息。
- -k<0 或 1>：重设硬盘时，保留-dmu 参数的设定。
- -K<0 或 1>：重设硬盘时，保留-APSWXZ 参数的设定。
- -m<分区数>：设定硬盘多重分区存取的分区数。
- -n<0 或 1>：忽略硬盘写入时所发生的错误。
- -p<PIO 模式>：设定硬盘的 PIO 模式。
- -P<分区数>：设定硬盘内部块区的分区数。
- -q：在执行后续的参数时，不在屏幕上显示任何信息。
- -r<0 或 1>：设定硬盘的读写模式。
- -S<时间>：设定硬盘进入省电模式前的等待时间。
- -t：评估硬盘的读取效率。
- -T：评估硬盘块区的读取效率。
- -u<0 或 1>：在硬盘存取时，允许其他中断要求同时执行。
- -v：显示硬盘的相关设定。
- -W<0 或 1>：设定硬盘的写入块区。
- -X<传输模式>：设定硬盘的传输模式。
- -y：使 IDE 硬盘进入省电模式。
- -Y：使 IDE 硬盘进入睡眠模式。
- -Z：关闭某些 Seagate 硬盘的自动省电功能。

### fsck

功能说明：检查文件系统并尝试修复错误。

语法：fsck [-aANPrRsTV][-t<文件系统类型>][文件系统...]

使用说明：当文件系统发生错误时，可用 fsck 指令尝试加以修复。

参数：

- -a：自动修复文件系统，不询问任何问题。

- -A：依照/etc/fstab 配置文件的内容，检查文件内所列的全部文件系统。
- -N：不执行指令，仅列出实际执行会进行的动作。
- -P：当搭配“-A”参数使用时，则会同时检查所有的文件系统。
- -r：采用互动模式，在执行修复时询问问题，让用户得以确认并决定处理方式。
- -R：当搭配“-A”参数使用时，则会略过/目录的文件系统不予检查。
- -s：依序执行检查作业，而非同时执行。
- -t<文件系统类型>：指定要检查的文件系统类型。
- -T：执行 fsck 指令时，不显示标题信息。
- -V：显示指令的执行过程。

## fsck.ext2

功能说明：检查文件系统并尝试修复错误。

语法：fsck.ext2 [-acdfFnprsStvVy][-b<分区第一个磁区地址>][-B<区块大小>][-C<反叙述器>][-I<inode 缓冲区块数>][-l/L<损坏区块文件>][-P<处理 inode 大小>][外围设备代号]

使用说明：当 ext2 文件系统发生错误时，可用 fsck.ext2 指令尝试加以修复。

参数：

- -a：自动修复文件系统，不询问任何问题。
- -b<分区第一个磁区地址>：指定分区的第一个磁区的起始地址，也就是 Super Block。
- -B<区块大小>：设置该分区每个区块的大小。
- -c：检查指定的文件系统内是否存在损坏的区块。
- -C<反叙述器>：指定反叙述器，fsck.ext2 指令会把全部的执行过程都交由其逆向叙述，便于排错或监控程序执行的情形。
- -d：详细显示指令的执行过程，便于排错或分析程序执行的情形。
- -f：强制对该文件系统进行完整检查，即使该文件系统在概略检查下没有问题。
- -F：检查文件系统之前，先清理该保存设备块区内的数据。
- -I<inode 缓冲区块数>：设置欲检查的文件系统，其 inode 缓冲区的区块数目。
- -l<损坏区块文件>：把文件中所列出的区块，视为损坏区块并将其标示出来，避免应用程序使用该区块。
- -L<损坏区块文件>：此参数的效果和指定“-l”参数类似，但在参考损坏区块文件标示损坏区块之前，会先将原来标示成损坏区块者统统清除，即全部重新设置，而非仅是加入新的损坏区块标示。
- -n：把欲检查的文件系统设成只读，并关闭互动模式，否决所有询问的问题。
- -p：此参数的效果和指定“-a”参数相同。
- -P<处理 inode 大小>：设置 fsck.ext2 指令所能处理的 inode 大小为多少。
- -r：此参数将忽略不予处理，仅负责解决兼容性的问题。
- -s：检查文件系统时，交换每对字节的内容。
- -S：此参数的效果和指定“-s”参数类似，但不论该文件系统是否已是标准位顺序，一律交换每对字节的内容。
- -t：显示 fsck.ext2 指令的时序信息。
- -v：详细显示指令的执行过程。

- -V：显示版本信息。
- -y：关闭互动模式，且同意所有询问的问题。

### fdisk

功能说明：磁盘分区。

语法：fdisk [-b<分区大小>][-uv][外围设备代号]

或：fdisk [-l][-b<分区大小>][-uv][外围设备代号...]

或：fdisk [-s<分区编号>]

使用说明：fdisk 是用来磁盘分区的程序，它采用传统的问答式界面，而非类似 DOS fdisk 的 cfdisk 互动式操作界面，因此在使用上较为不便，但功能却丝毫不打折扣。

参数：

- -b<分区大小>：指定每个分区的大小。
- -l：列出指定的外围设备的分区表状况。
- -s<分区编号>：将指定的分区大小输出到标准输出上，单位为区块。
- -u：搭配“-l”参数列表，会用分区数目取代柱面数目来表示每个分区的起始地址。
- -v：显示版本信息。

### e2fsck

功能说明：检查 ext2 文件系统的正确性。

语法：e2fsck [-acCdfFnprsStvVy][-b<superblock>][-B<区块大小>][-l<文件>][-L<文件>][设备名称]

使用说明：e2fsck 执行后的传回值及代表意义如下。

- 0：没有任何错误发生。
- 1：文件系统发生错误，并且已经修正。
- 2：文件系统发生错误，并且已经修正。
- 4：文件系统发生错误，但没有修正。
- 8：运作时发生错误。
- 16：使用的语法发生错误。
- 128：共享的函数库发生错误。

参数：

- -a：不询问使用者意见便自动修复文件系统。
- -b<superblock>：指定 superblock，而不使用预设的 superblock。
- -B<区块大小>：指定区块的大小，单位为字节。
- -c：一并执行 badblocks，以标示损坏的区块。
- -C：将检查过程的信息完整记录在 file descriptor 中，使得整个检查过程都能完整监控。
- -d：显示排错信息。
- -f：即使文件系统没有错误迹象，仍强制地检查正确性。
- -F：执行前先清除设备的缓冲区。
- -l<文件>：将文件中指定的区块加到损坏区块列表。

- -L<文件>：先清除损坏区块列表，再将文件中指定的区块加到损坏区块列表。因此损坏区块列表的区块跟文件中指定的区块是一样的。
- -n：以只读模式开启文件系统，并采取非互动方式执行，所有的问题对话均设置以“no”回答。
- -p：不询问使用者意见，便自动修复文件系统。
- -r：此参数只为了兼容性而存在，并无实际作用。
- -s：如果文件系统的字节顺序不适当，就交换字节顺序，否则不做任何动作。
- -S：不管文件系统的字节顺序，一律交换字节顺序。
- -t：显示时间信息。
- -v：执行时显示详细的信息。
- -V：显示版本信息。
- -y：采取非互动方式执行，所有的问题均设置以“yes”回答。

## dd

功能说明：读取、转换并输出数据。

语法：dd [bs=<字节数>][cbs=<字节数>][conv=<关键字>][count=<区块数>][ibs=<字节数>][if=<文件>][obs=<字节数>][of=<文件>][seek=<区块数>][skip=<区块数>][--help][--version]

使用说明：dd 可从标准输入或文件读取数据，依指定的格式来转换数据，再输出到文件、设备或标准输出。

参数：

- bs=<字节数>：将 ibs（输入）与 obs（输出）设成指定的字节数。
- cbs=<字节数>：转换时，每次只转换指定的字节数。
- conv=<关键字>：指定文件转换的方式。
- count=<区块数>：仅读取指定的区块数。
- ibs=<字节数>：每次读取的字节数。
- if=<文件>：从文件读取。
- obs=<字节数>：每次输出的字节数。
- of=<文件>：输出到文件。
- seek=<区块数>：一开始输出时，跳过指定的区块数。
- skip=<区块数>：一开始读取时，跳过指定的区块数。
- --help：帮助。
- --version：显示版本信息。

## badblocks

功能说明：检查磁盘装置中损坏的区块。

语法：badblocks [-svw][-b<区块大小>][-o<输出文件>][磁盘装置][磁盘区块数][启始区块]

使用说明：执行指令时须指定所要检查的磁盘装置，及此装置的磁盘区块数。

参数：

- -b<区块大小>：指定磁盘的区块大小，单位为字节。
- -o<输出文件>：将检查的结果写入指定的输出文件。

- -s：在检查时显示进度。
- -v：执行时显示详细的信息。
- -w：在检查时，执行写入测试。
- 磁盘装置：指定要检查的磁盘装置。
- 磁盘区块数：指定磁盘装置的区块总数。
- 启始区块：指定要从哪个区块开始检查。

## ar

功能说明：建立或修改备存文件，或是从备存文件中抽取文件。

语法：ar[-dmpqrtx][cfosSuvV][a<成员文件>][b<成员文件>][i<成员文件>][备存文件][成员文件]

使用说明：ar 可让您集合许多文件，成为单一的备存文件。在备存文件中，所有成员文件皆保有原来的属性与权限。

指令参数：

- -d：删除备存文件中的成员文件。
- -m：变更成员文件在备存文件中的次序。
- -p：显示备存文件中的成员文件内容。
- -q：将文件附加在备存文件末端。
- -r：将文件插入备存文件中。
- -t：显示备存文件中所包含的文件。
- -x：自备存文件中取出成员文件。

选项参数：

- a<成员文件>：将文件插入备存文件中指定的成员文件之后。
- b<成员文件>：将文件插入备存文件中指定的成员文件之前。
- c：建立备存文件。
- f：为避免过长的文件名不兼容于其他系统的 ar 指令，因此可利用此参数，截掉文件名。

## bunzip2

功能说明：.bz2 文件的解压缩程序。

语法：bunzip2 [-fkLsvV][.bz2 压缩文件]

使用说明：bunzip2 可解压缩.bz2 格式的压缩文件。bunzip2 实际上是 bzip2 的符号连接，执行 bunzip2 与 bzip2-d 的效果相同。

参数：

- -f 或--force：解压缩时，若输出的文件与现有文件同名时，预设不会覆盖现有的文件。若要覆盖，请使用此参数。
- -k 或--keep：在解压缩后，预设会删除原来的压缩文件。若要保留压缩文件，请使用此参数。
- -s 或--small：降低程序执行时内存的使用量。
- -v 或--verbose：解压缩文件时显示详细的信息。
- -l,--license,-V 或--version：显示版本信息。

## bzip2

功能说明：.bz2 文件的压缩程序。

语法：bzip2 [-cdfhkLstvVz][--repetitive-best][--repetitive-fast][-压缩等级][要压缩的文件]

使用说明：bzip2 采用新的压缩演算法，压缩效果比传统的 LZ77/LZ78 压缩演算法来得好。若没有加上任何参数，bzip2 压缩完文件后会产生.bz2 的压缩文件，并删除原始的文件。

参数：

- -c 或--stdout：将压缩与解压缩的结果送到标准输出。
- -d 或--decompress：执行解压缩。
- -f 或--force：bzip2 在压缩或解压缩时，若输出文件与现有文件同名，预设不会覆盖现有文件。若要覆盖，请使用此参数。
- -h 或--help：显示帮助。
- -k 或--keep：bzip2 在压缩或解压缩后，会删除原始的文件。若要保留原始文件，请使用此参数。
- -s 或--small：降低程序执行时内存的使用量。
- -t 或--test：测试.bz2 压缩文件的完整性。
- -v 或--verbose：压缩或解压缩文件时，显示详细的信息。
- -z 或--compress：强制执行压缩。
- -L,--license：显示 bzip2 的许可证书版本
- -V 或--version：显示版本信息。
- --repetitive-best：若文件中有重复出现的资料时，可利用此参数提高压缩效果。
- --repetitive-fast：若文件中有重复出现的资料时，可利用此参数加快执行速度。
- -压缩等级：压缩时的区块大小。

## bzip2recover

功能说明：用来修复损坏的.bz2 文件。

语法：bzip2recover [.bz2 压缩文件]

使用说明：bzip2 是以区块的方式来压缩文件，每个区块视为独立的单位。因此，当某一区块损坏时，便可利用 bzip2recover 试着将文件中的区块隔开来，以便解压缩正常的区块。通常只适用在压缩文件很大的情况。

## compress

功能说明：压缩或解压文件。

语法：compress [-cdfrvV][-b<压缩效率>][文件或目录...]

使用说明：compress 是个历史悠久的压缩程序，文件经它压缩后，其名称后面会多出“.Z”的扩展名。当要解压缩时，可执行 uncompress 指令。事实上 uncompress 是指向 compress 的符号连接，因此不论是压缩或解压缩，都可通过 compress 指令单独完成。

参数：

- -b<压缩效率>：压缩效率是一个介于 9～16 的数值，预设值为“16”，指定愈大的数值，压缩效率就愈高。

- -c：把压缩后的文件输出到标准输出设备，不去更动原始文件。
- -d：对文件进行解压缩而非压缩。
- -f：强制保存压缩文件，不理会文件名称或硬连接是否存在、该文件是否为符号连接以及压缩效率高低的问题。
- -r：递归处理，将指定目录下的所有文件及子目录一并处理。
- -v：显示指令的执行过程。
- -V：显示指令的版本及程序预设值。

## cpio

功能说明：备份文件。

语法：cpio [-0aABckLovV][-C<输入/输出大小>][-F<备份档>][-H<备份格式>][-O<备份档>][--block-size=<区块大小>][--force-local][--help][--quiet][--version]

或：cpio [-bBcdfikmnrsStuvV][-C<输入/输出大小>][-E<范本文件>][-F<备份档>][-H<备份格式>][-I<备份档>][-M<回传信息>][-R<拥有者><:/.><所属群组>][--block-size=<区块大小>][--force-local][--help][--no-absolute-filenames][--no-preserve-owner][--only-verify-crc][--quiet][--sparse][--version][范本样式...]

或：cpio [-0adkiLmpuvV][-R<拥有者><:/.><所属群组>][--help][--no-preserve-owner][--quiet][--sparse][--version][目的目录]

使用说明：cpio 是用来建立、还原备份档的工具程序，它可以加入、解开 cpio 或 tra 备份档内的文件。

参数：

- -0 或--null：接受新增列控制字符，通常配合 find 指令的“-print0”参数使用。
- -a 或--reset-access-time：重新设置文件的存取时间。
- -A 或--append：附加到已存在的备份档中，且这个备份档必须存放在磁盘上，而不能放置于磁带机里。
- -b 或--swap：此参数的效果和同时指定“-sS”参数相同。
- -B：将输入/输出的区块大小改成 5210 字节。
- -c：使用旧 ASCII 备份格式。
- -C<输入/输出大小>或--io-size=<区块大小>：设置输入/输出的区块大小，单位是字节。
- -d 或--make-directories：如有需要，cpio 会自行建立目录。
- -E<范本文件>或--pattern-file=<范本文件>：指定范本文件，其内含有一个或多个范本样式，让 cpio 解开符合范本条件的文件，格式为每列一个范本样式。
- -f 或--nonmatching：让 cpio 解开所有不符合范本条件的文件。
- -F<备份档>或--file=<备份档>：指定备份档的名称，用来取代标准输入/输出，也能借此通过网络使用另一台主机的保存设备存取备份档。
- -H<备份格式>：指定备份时欲使用的文件格式。
- -i 或--extract：执行 copy-in 模式，还原备份档。
- -I<备份档>：指定备份档的名称，用来取代标准输入，也能借此通过网络使用另一台主机的保存设备读取备份档。

- -k：此参数将忽略不予处理，仅负责解决 cpio 不同版本间的兼容性问题。
- -l 或--link：以硬连接的方式取代复制文件，可在 copy-pass 模式下运用。
- -L 或--dereference：不建立符号连接，直接复制该连接所指向的原始文件。
- -m 或 preserve-modification-time：不去更换文件的更改时间。
- -M<回传信息>或--message=<回传信息>：设置更换保存媒体的信息。
- -n 或--numeric-uid-gid：使用“-tv”参数列出备份档的内容时，若再加上参数“-n”，则会以用户识别码和群组识别码替代拥有者和群组名称列出文件清单。
- -o 或--create：执行 copy-out 模式，建立备份档。
- -O<备份档>：指定备份档的名称，用来取代标准输出，也能借此通过网络使用另一台主机的保存设备存放备份档。
- -p 或--pass-through：执行 copy-pass 模式，略过备份步骤，直接将文件复制到目的目录。
- -r 或--rename：当有文件名称需要改动时，采用互动模式。
- -R<拥有者><:/.><所属群组>或----owner<拥有者><:/.><所属群组>：在 copy-in 模式还原备份档，或 copy-pass 模式复制文件时，可指定这些备份、复制的文件的拥有者与所属群组。
- -s 或--swap-bytes：交换每对字节的内容。
- -S 或--swap-halfwords：交换每半个字节的内容。
- -t 或--list：将输入的内容呈现出来。
- -u 或--unconditional：置换所有文件，不论日期时间的新旧与否，皆不予询问而直接覆盖。
- -v 或--verbose：详细显示指令的执行过程。
- -V 或--dot：执行指令时，在每个文件的执行程序前面加上“.”号。
- --block-size=<区块大小>：设置输入/输出的区块大小，假如设置数值为 5，则区块大小为 2500，若设置成 10，则区块大小为 5120，以此类推。
- --force-local：强制将备份档存放在本地主机。
- --help：在线帮助。
- --no-absolute-filenames：使用相对路径建立文件名称。
- --no-preserve-owner：不保留文件的拥有者，谁解开了备份档，那些文件就归谁所有。
- -only-verify-crc：当备份档采用 CRC 备份格式时，可使用这项参数检查备份档内的每个文件是否正确无误。
- --quiet：不显示复制了多少区块。
- --sparse：倘若一个文件内含大量的连续 0 字节，则将此文件存成稀疏文件。
- --version：显示版本信息。

## dump

功能说明：备份文件系统。

语法：dump [-cnu][-0123456789][-b<区块大小>][-B<区块数目>][-d<密度>][-f<设备名称>][-h<层级>][-s<磁带长度>][-T<日期>][目录或文件系统]

或：dump [-wW]

使用说明：dump 为备份工具程序，可将目录或整个文件系统备份至指定的设备，或备份成一个大文件。

参数：

- -0123456789：备份的层级。
- -b<区块大小>：指定区块的大小，单位为 KB。
- -B<区块数目>：指定备份卷册的区块数目。
- -c：修改备份磁带预设的密度与容量。
- -d<密度>：设置磁带的密度，单位为 BPI。
- -f<设备名称>：指定备份设备。
- -h<层级>：当备份层级等于或大于指定的层级时，将不备份用户标示为“nodump”的文件。
- -n：当备份工作需要管理员介入时，向所有“operator”群组中的使用者发出通知。
- -s<磁带长度>：备份磁带的长度，单位为英尺。
- -T<日期>：指定开始备份的时间与日期。
- -u：备份完毕后，在/etc/dumpdates 中记录备份的文件系统、层级、日期与时间等。
- -w：与-W 类似，但仅显示需要备份的文件。
- -W：显示需要备份的文件及其最后一次备份的层级、时间与日期。

## gunzip

功能说明：解压文件。

语法：gunzip [-acfhlLnNqrtvV][-S<压缩字尾字符串>][文件...]

或：gunzip [-acfhlLnNqrtvV][-s<压缩字尾字符串>][目录]

使用说明：gunzip 是一个使用广泛的解压缩程序，它用于解开被 gzip 压缩过的文件，这些压缩文件预设最后的扩展名为“.gz”。事实上 gunzip 就是 gzip 的硬连接，因此不论是压缩或解压缩，都可通过 gzip 指令单独完成。

参数：

- -a 或--ascii：使用 ASCII 文字模式。
- -c 或--stdout 或--to-stdout：把解压后的文件输出到标准输出设备。
- -f 或-force：强行解开压缩文件，不理会文件名称或硬连接是否存在以及该文件是否为符号连接。
- -h 或--help：在线帮助。
- -l 或--list：列出压缩文件的相关信息。
- -L 或--license：显示版本与版权信息。
- -n 或--no-name：解压缩时，若压缩文件内含有原来的文件名称及时间戳记，则将其忽略不予处理。
- -N 或--name：解压缩时，若压缩文件内含有原来的文件名称及时间戳记，则将其回存到解开的文件上。
- -q 或--quiet：不显示警告信息。
- -r 或--recursive：递归处理，将指定目录下的所有文件及子目录一并处理。
- -s<压缩字尾字符串>或--suffix<压缩字尾字符串>：更改压缩字尾字符串。
- -t 或--test：测试压缩文件是否正确无误。
- -v 或--verbose：显示指令的执行过程。

- -V 或--version：显示版本信息。

## gzexe

功能说明：压缩执行文件。

语法：gzexe [-d][执行文件...]

使用说明：gzexe 是用来压缩执行文件的程序。当您去执行被压缩过的执行文件时，该文件会自动解压，然后继续执行，和使用一般的执行文件相同。

参数：

- -d：解开压缩文件。

## gzip

功能说明：压缩文件。

语法：gzip [-acdfhlLnNqrtvV][-S<压缩字尾字符串>][-<压缩效率>][--best/fast][文件...]

或：gzip [-acdfhlLnNqrtvV][-S<压缩字尾字符串>][-<压缩效率>][--best/fast][目录]

使用说明：gzip 是一个使用广泛的压缩程序，文件经它压缩过后，其名称后面会多出“.gz”的扩展名。

参数：

- -a 或--ascii：使用 ASCII 文字模式。
- -c 或--stdout 或--to-stdout：把压缩后的文件输出到标准输出设备，不去改动原始文件。
- -d 或--decompress 或----uncompress：解开压缩文件。
- -f 或--force：强行压缩文件。不理会文件名称或硬连接是否存在以及该文件是否为符号连接。
- -h 或--help：在线帮助。
- -l 或--list：列出压缩文件的相关信息。
- -L 或--license：显示版本与版权信息。
- -n 或--no-name：压缩文件时，不保存原来的文件名称及时间戳记。
- -N 或--name：压缩文件时，保存原来的文件名称及时间戳记。
- -q 或--quiet：不显示警告信息。
- -r 或--recursive：递归处理，将指定目录下的所有文件及子目录一并处理。
- -S<压缩字尾字符串>或----suffix<压缩字尾字符串>：更改压缩字尾字符串。
- -t 或--test：测试压缩文件是否正确无误。
- -v 或--verbose：显示指令的执行过程。
- -V 或--version：显示版本信息。
- -<压缩效率>：压缩效率是一个介于 1～9 的数值，预设值为“6”，指定愈大的数值，压缩效率就会愈高。
- --best：此参数的效果和指定“-9”参数相同。
- --fast：此参数的效果和指定“-1”参数相同。

## lha

功能说明：压缩或解压缩文件。

语法：lha [-acdfglmnpqtuvx][-a<0/1/2>/u</0/1/2>][-<a/c/u>d][-<e/x>i][-<a/u>o][-<e/x>w=<目的目录>][-<a/u>z][压缩文件][文件…]

或：lha [-acdfglmnpqtuvx][-a<0/1/2>/u</0/1/2>][-<a/c/u>d][-<e/x>i][-<a/u>o][-<e/x>w=<目的目录>][-<a/u>z][压缩文件][目录...]

使用说明：lha 是从 lharc 演变而来的压缩程序，文件经它压缩后，会另外产生具有“.lzh”扩展名的压缩文件。

参数：

- -a 或 a：压缩文件，并加入到压缩文件内。
- -a<0/1/2>/u</0/1/2>：压缩文件时，采用不同的文件头。
- -c 或 c：压缩文件，重新构建新的压缩文件后，再将其加入。
- -d 或 d：从压缩文件内删除指定的文件。
- -<a/c/u>d 或<a/c/u>d：压缩文件，然后将其加入，重新构建、更新压缩文件或删除原始文件，也就是把文件移到压缩文件中。
- -e 或 e：解开压缩文件。
- -f 或 f：强制执行 lha 命令，在解压时会直接覆盖已有的文件而不加以询问。
- -g 或 g：使用通用的压缩格式，便于解决兼容性问题。
- -<e/x>i 或<e/x>i：解开压缩文件时，忽略保存在压缩文件内的文件路径，直接将其解压后存放在现行目录下或是指定的目录中。
- -l 或 l：列出压缩文件的相关信息。
- -m 或 m：此参数的效果和同时指定“-ad”参数相同。
- -n 或 n：不执行指令，仅列出实际执行会进行的动作。
- -<a/u>o 或<a/u>o：采用 lharc 兼容格式，将压缩后的文件加入，更新压缩文件。
- -p 或 p：从压缩文件内输出到标准输出设备。
- -q 或 q：不显示指令执行过程。
- -t 或 t：检查备份文件内的每个文件是否正确无误。
- -u 或 u：更换较新的文件到压缩文件内。
- -u</0/1/2>或 u</0/1/2>：在文件压缩时采用不同的文件头，然后更新到压缩文件内。
- -v 或 v：详细列出压缩文件的相关信息。
- -<e/x>w=<目的目录>或<e/x>w=<目的目录>：指定解压缩的目录。
- -x 或 x：解开压缩文件。
- -<a/u>z 或<a/u>z：不压缩文件，直接把它加入，更新压缩文件。

## restore

功能说明：还原（Restore）由倾倒（Dump）操作所备份下来的文件或整个文件系统（一个分区）。

语法：restore [-cCvy][-b<区块大小>][-D<文件系统>][-f<备份文件>][-s<文件编号>]

或：restore [-chimvy][-b<区块大小>][-f<备份文件>][-s<文件编号>]

或：restore [-crvy][-b<区块大小>][-f<备份文件>][-s<文件编号>]

或：restore [-cRvy][-b<区块大小>][-D<文件系统>][-f<备份文件>][-s<文件编号>]

或：restore [chtvy][-b<区块大小>][-D<文件系统>][-f<备份文件>][-s<文件编号>][文件...]

或：restore [-chmvxy][-b<区块大小>][-D<文件系统>][-f<备份文件>][-s<文件编号>][文件...]

使用说明：restore 指令所进行的操作和 dump 指令相反，倾倒操作可用来备份文件，而还原操作则是写回这些已备份的文件。

参数：

- -b<区块大小>：设置区块大小，单位是字节。
- -c：不检查倾倒操作的备份格式，仅准许读取使用旧格式的备份文件。
- -C：使用对比模式，将备份的文件与现行的文件相互对比。
- -D<文件系统>：允许用户指定文件系统的名称。
- -f<备份文件>：从指定的文件中读取备份数据，进行还原操作。
- -h：仅解出目录而不包括与该目录相关的所有文件。
- -i：使用互动模式，在进行还原操作时，restore 指令将依序询问用户。
- -m：解开符合指定的 inode 编号的文件或目录而非采用文件名称指定。
- -r：进行还原操作。
- -R：全面还原文件系统时，检查应从何处开始进行。
- -s<文件编号>：当备份数据超过一卷磁带时，您可以指定备份文件的编号。
- -t：指定文件名称，若该文件已存在备份文件中，则列出它们的名称。
- -v：显示指令执行过程。
- -x：设置文件名称，且从指定的存储媒体里读入它们，若该文件已存在备份文件中，则将其还原到文件系统内。
- -y：不询问任何问题，一律以同意回答并继续执行指令。

## tar

功能说明：备份文件。

语法：tar [-ABcdgGhiklmMoOpPrRsStuUvwWxzZ][-b<区块数目>][-C<目的目录>][-f<备份文件>][-F<Script 文件>][-K<文件>][-L<媒体容量>][-N<日期时间>][-T<范本文件>][-V<卷册名称>][-X<范本文件>][-<设备编号><存储密度>][--after-date=<日期时间>][--atime-preserve][--backup=<备份方式>][--checkpoint][--concatenate][--confirmation][--delete][--exclude=<范本样式>][--force-local][--group=<群组名称>][--help][--ignore-failed-read][--new-volume-script=<Script 文件>][--newer-mtime][--no-recursion][--null][--numeric-owner][--owner=<用户名称>][--posix][--erve][--preserve-order][--preserve-permissions][--record-size=<区块数目>][--recursive-unlink][--remove-files][--rsh-command=<执行指令>][--same-owner][--suffix=<备份字尾字符串>][--totals][--use-compress-program=<执行指令>][--version][--volno-file=<编号文件>][文件或目录...]

使用说明：tar 是用来建立、还原备份文件的工具程序，它可以加入，解开备份文件内的文件。

参数：

- -A 或--catenate：新增文件到已存在的备份文件。
- -b<区块数目>或--blocking-factor=<区块数目>：设置每笔记录的区块数目，每个区块大小为 12 字节。

- -B 或--read-full-records：读取数据时重设区块大小。
- -c 或--create：建立新的备份文件。
- -C<目的目录>或--directory=<目的目录>：切换到指定的目录。
- -d 或--diff 或--compare：对比备份文件内和文件系统上的文件差异。
- -f<备份文件>或--file=<备份文件>：指定备份文件。
- -F<Script 文件>或--info-script=<Script 文件>：每次更换磁带时，就执行指定的 Script 文件。
- -g 或--listed-incremental：处理 GNU 格式的大量备份。
- -G 或--incremental：处理旧的 GNU 格式的大量备份。
- -h 或--dereference：不建立符号连接，直接复制该连接所指向的原始文件。
- -i 或--ignore-zeros：忽略备份文件中的 0 字节区块，也就是 EOF。
- -k 或--keep-old-files：解开备份文件时，不覆盖已有的文件。
- -K<文件>或--starting-file=<文件>：从指定的文件开始还原。
- -l 或--one-file-system：复制的文件或目录存放的文件系统，必须与 tar 指令执行时所处的文件系统相同，否则不予复制。
- -L<媒体容量>或-tape-length=<媒体容量>：设置存放媒体的容量，单位以 1024 字节计算。
- -m 或--modification-time：还原文件时，不变更文件的更改时间。
- -M 或--multi-volume：在建立、还原备份文件或列出其中的内容时，采用多卷册模式。
- -N<日期时间>或--newer=<日期时间>：只将此指定日期更新的文件保存到备份文件里。
- -o 或--old-archive 或--portability：将资料写入备份文件时使用 V7 格式。
- -O 或--stdout：把从备份文件里还原的文件输出到标准输出设备。
- -p 或--same-permissions：用原来的文件权限还原文件。
- -P 或--absolute-names：文件名使用绝对名称，不移除文件名称前的“/”号。
- -r 或--append：新增文件到已存在的备份文件的结尾部分。
- -R 或--block-number：列出每个信息在备份文件中的区块编号。
- -s 或--same-order：还原文件的顺序和备份文件内的存放顺序相同。
- -S 或--sparse：倘若一个文件内含大量的连续 0 字节，则将此文件存成稀疏文件。
- -t 或--list：列出备份文件的内容。
- -T<范本文件>或--files-from=<范本文件>：指定范本文件，其内含有一个或多个范本样式，让 tar 解开或建立符合设置条件的文件。
- -u 或--update：仅置换较备份文件内的文件更新的文件。
- -U 或--unlink-first：解开压缩文件还原文件之前，先解除文件的连接。
- -v 或--verbose：显示指令的执行过程。
- -V<卷册名称>或--label=<卷册名称>：建立使用指定的卷册名称的备份文件。
- -w 或--interactive：遭遇问题时先询问用户。
- -W 或--verify：写入备份文件后，确认文件正确无误。
- -x 或--extract 或--get：从备份文件中还原文件。
- -X<范本文件>或--exclude-from=<范本文件>：指定范本文件，其内含有一个或多个范本样式，让 ar 排除符合设置条件的文件。
- -z 或--gzip 或--ungzip：通过 gzip 指令处理备份文件。
- -Z 或--compress 或--uncompress：通过 compress 指令处理备份文件。

- -<设备编号><存储密度>：设置备份用的外围设备编号及存放数据的密度。
- --after-date=<日期时间>：此参数的效果和指定“-N”参数相同。
- --atime-preserve：不变更文件的存取时间。
- --backup=<备份方式>或--backup：移除文件前先进行备份。
- --checkpoint：读取备份文件时列出目录名称。
- --concatenate：此参数的效果和指定“-A”参数相同。
- --confirmation：此参数的效果和指定“-w”参数相同。
- --delete：从备份文件中删除指定的文件。
- --exclude=<范本样式>：排除符合范本样式的文件。
- --group=<群组名称>：把加入设备文件中的所属群组设成指定的群组。
- --help：在线帮助。
- --ignore-failed-read：忽略数据读取错误，不中断程序的执行。
- --new-volume-script=<Script 文件>：此参数的效果和指定“-F”参数相同。
- --newer-mtime：只保存更改过的文件。
- --no-recursion：不做递归处理，也就是指定目录下的所有文件及子目录不予处理。
- --null：从 null 设备读取文件名称。
- --numeric-owner：以用户识别码及群组识别码取代用户名称和群组名称。
- --owner=<用户名称>：把加入备份文件中的拥有者设成指定的用户。
- --posix：将数据写入备份文件时使用 POSIX 格式。
- --erve：：此参数的效果和指定“-ps”参数相同。
- --preserve-order：此参数的效果和指定“-A”参数相同。
- --preserve-permissions：此参数的效果和指定“-p”参数相同。
- --record-size=<区块数目>：此参数的效果和指定“-b”参数相同。
- --recursive-unlink：解开压缩文件还原目录之前，先解除整个目录下所有文件的连接。
- --remove-files：文件加入备份文件后，就将其删除。
- --rsh-command=<执行指令>：设置要在远端主机上执行的指令，以取代 rsh 指令。
- --same-owner：尝试以相同的文件拥有者还原文件。
- --suffix=<备份字尾字符串>：移除文件前先行备份。
- --totals：备份文件建立后，列出文件大小。
- --use-compress-program=<执行指令>：通过指定的指令处理备份文件。
- --version：显示版本信息。
- --volno-file=<编号文件>：使用指定文件内的编号取代预设的卷册编号。

## unarj

功能说明：解压缩.arj 文件。

语法：unarj [eltx][.arj 压缩文件]

使用说明：unarj 为.arj 压缩文件的压缩程序。

参数：

- e：解压缩.arj 文件。
- l：显示压缩文件内所包含的文件。

- t：检查压缩文件是否正确。
- x：解压缩时保留原有的路径。

**unzip**

功能说明：解压缩 zip 文件。

语法：unzip [-cflptuvz][-abCjLMnoqsVX][-P<密码>][.zip 文件][文件][-d<目录>][-x<文件>]

或：unzip [-Z]

使用说明：unzip 为.zip 压缩文件的解压缩程序。

参数：

- -c：将解压缩的结果显示到屏幕上，并对字符做适当的转换。
- -f：更新现有的文件。
- -l：显示压缩文件内所包含的文件。
- -p：与-c 参数类似，会将解压缩的结果显示到屏幕上，但不会执行任何的转换。
- -t：检查压缩文件是否正确。
- -u：与-f 参数类似，但是除了更新现有的文件外，也会将压缩文件中的其他文件解压缩到目录中。
- -v：执行是时显示详细的信息。
- -z：仅显示压缩文件的备注文字。
- -a：对文本文件进行必要的字符转换。
- -b：不要对文本文件进行字符转换。
- -C：压缩文件中的文件名称区分大小写。
- -j：不处理压缩文件中原有的目录路径。
- -L：将压缩文件中的全部文件名改为小写。
- -M：将输出结果送到 more 程序处理。
- -n：解压缩时不要覆盖原有的文件。
- -o：不必先询问用户，unzip 执行后覆盖原有文件。
- -P<密码>：使用 zip 的密码选项。
- -q：执行时不显示任何信息。
- -s：将文件名中的空白字符转换为下划线字符。
- -V：保留 VMS 的文件版本信息。
- -X：解压缩时同时回存文件原来的 UID/GID。
- .zip 文件：指定.zip 压缩文件。
- 文件：指定要处理.zip 压缩文件中的哪些文件。
- -d<目录>：指定文件解压缩后所要存储的目录。
- -x<文件>：指定不要处理.zip 压缩文件中的哪些文件。
- -Z：unzip-Z 等于执行 zipinfo 指令。

**zip**

功能说明：压缩文件。

语法：zip [-AcdDfFghjJKlLmoqrSTuvVwXyz$][-b<工作目录>][-ll][-n<字尾字符串>][-t<日期

时间>][-<压缩效率>][压缩文件][文件...][-i<范本样式>][-x<范本样式>]

使用说明：zip 是个使用广泛的压缩程序，文件经它压缩后会另外产生具有“.zip”扩展名的压缩文件。

参数：

- -A：调整可执行的自动解压缩文件。
- -b<工作目录>：指定暂时存放文件的目录。
- -c：替每个被压缩的文件加上注释。
- -d：从压缩文件内删除指定的文件。
- -D：压缩文件内不建立目录名称。
- -f：此参数的效果和指定“-u”参数类似，但不仅更新既有文件，如果某些文件原本不存在于压缩文件内，使用本参数会一并将其加入压缩文件中。
- -F：尝试修复已损坏的压缩文件。
- -g：将文件压缩后附加在既有的压缩文件之后，而非另行建立新的压缩文件。
- -h：在线帮助。
- -i<范本样式>：只压缩符合条件的文件。
- -j：只保存文件名称及其内容，而不存放任何目录名称。
- -J：删除压缩文件前面不必要的数据。
- -K：使用 MS-DOS 兼容格式的文件名称。
- -l：压缩文件时，把 LF 字符置换成 LF+CR 字符。
- -ll：压缩文件时，把 LF+CR 字符置换成 LF 字符。
- -L：显示版权信息。
- -m：将文件压缩并加入压缩文件后，删除原始文件，即把文件移到压缩文件中。
- -n<字尾字符串>：不压缩具有特定字尾字符串的文件。
- -o：以压缩文件内拥有最新更改时间的文件为准，将压缩文件的更改时间设成和该文件相同。
- -q：不显示指令的执行过程。
- -r：递归处理，将指定目录下的所有文件和子目录一并处理。
- -S：包含系统和隐藏文件。
- -t<日期时间>：把压缩文件的日期设成指定的日期。
- -T：检查备份文件内的每个文件是否正确无误。
- -u：更换较新的文件到压缩文件内。
- -v：显示指令的执行过程或显示版本信息。
- -V：保存 VMS 操作系统的文件属性。
- -w：在文件名称里加入版本编号，本参数仅在 VMS 操作系统下有效。
- -x<范本样式>：压缩时排除符合条件的文件。
- -X：不保存额外的文件属性。
- -y：直接保存符号连接，而非该连接所指向的文件，本参数仅在 UNIX 之类的系统下有效。
- -z：替压缩文件加上注释。
- -$：保存第一个被压缩文件所在磁盘的卷册名称。
- -<压缩效率>：压缩效率是一个介于 1～9 的数值。

### zipinfo

功能说明：列出压缩文件信息。

语法：zipinfo [-12hlmMstTvz][压缩文件][文件...][-x<范本样式>]

使用说明：执行 zipinfo 指令可得知 zip 压缩文件的详细信息。

参数：

- -1：只列出文件名称。
- -2：此参数的效果和指定-1 参数类似，但可搭配-h、-t 和-z 参数使用。
- -h：只列出压缩文件的文件名称。
- -l：此参数的效果和指定-m 参数类似，但会列出原始文件的大小而非每个文件的压缩率。
- -m：此参数的效果和指定-s 参数类似，但多会列出每个文件的压缩率。
- -M：若信息内容超过一个画面，则采用类似 more 指令的方式列出信息。
- -s：用类似执行“ls-l”指令的效果列出压缩文件内容。
- -t：只列出压缩文件内所包含的文件数目、压缩前后的文件大小及压缩率。
- -T：将压缩文件内每个文件的日期时间用年、月、日、时、分、秒的顺序列出。
- -v：详细显示压缩文件内每一个文件的信息。
- -x<范本样式>：不列出符合条件的文件的信息。
- -z：如果压缩文件内含有注释，就将注释显示出来。

## A.5 网络命令

### arpwatch

功能说明：监听网络上 ARP 的记录。

语法：arpwatch [-d][-f<记录文件>][-i<接口>][-r<记录文件>]

使用说明：ARP（Address Resolution Protocol）是用来解析 IP 与网络装置硬件地址的协议。arpwatch 可监听区域网络中的 ARP 数据包并记录，同时将监听到的变化通过 E-mail 来报告。

参数：

- -d：启动排错模式。
- -f<记录文件>：设置存储 ARP 记录的文件，预设为/var/arpwatch/arp.dat。
- -i<接口>：指定监听 ARP 的接口，预设的接口为 eth0。
- -r<记录文件>：从指定的文件中读取 ARP 记录，而不是从网络上监听。

### cu

功能说明：连接另一个系统主机。

语法：cu [dehnotv][-a<通信端口>][-c<电话号码>][-E<脱离字符>][-I<设置文件>][-l<外围设备代号>][-s<连线速率>][-x<排错模式>][-z<系统主机>][--help][-nostop][--parity=none][<系统主机>/<电话号码>]

使用说明：本指令可连接另一台主机，并采用类似拨号终端机的接口工作，也可执行简易的文件传输作业。

参数：

- -a<通信端口>或-p<通信端口>或--port<通信端口>：使用指定的通信端口进行连线。
- -c<电话号码>或--phone<电话号码>：拨打该电话号码。
- -d：进入排错模式。
- -e 或--parity=even：使用双同位检查。
- -E<脱离字符>或--escape<脱离字符>：设置脱离字符。
- -h 或--halfduple：使用半双工模式。
- -I<设置文件>或--config<配置文件>：指定要使用的配置文件。
- -l<外围设备代号>或--line<外围设备代号>：指定某项外围设备，并作为连接的设备。
- -n 或--prompt：拨号时等待用户输入电话号码。
- -o 或--parity=odd：使用单同位检查。
- -s<连线速率>或--speed<连线速率>或--baud<连线速率>或-<连线速率>：设置连线的速率，单位以波特率计算。
- -t 或--maper：把 CR 字符置换成 LF+CR 字符。
- -v 或--version：显示版本信息。
- -x<排错模式>或--debug<排错模式>：使用排错模式。
- -z<系统主机>或--system<系统主机>：连接该系统主机。
- --help：在线帮助。
- --nostop：关闭 Xon/Xoff 软件流量控制。
- --parity=none：不使用同位检查。

## dnsconf

功能说明：设置 DNS 服务器组态。

语法：dnsconf [--deldomain<域>][--delsecondary<域>][--newdomain<域>][--set<主机><IP>][--setcname<CNAME><主机>][--setmx<域><主机>][--setns<域><主机>][--unset<主机>]

使用说明：dnsconf 实际上为 linuxconf 的符号连接，提供图形界截面的操作方式，供管理员管理 DNS 服务器。

参数：

- --deldomain<域>：删除域。
- --delsecondary<域>：删除次级域。
- --newdomain<域>：新增域。
- --set<主机><IP>：新增主机记录。
- --setcname<CNAME><主机>：设置“CNAME”。
- --setmx<域><主机>：指定域的邮件主机。
- --setns<域><主机>：指定域的 DNS 服务器。
- --unset<主机>：删除 DNS 中某台主机的记录。

## getty

功能说明：设置终端机模式、连线速率和管制线路。

语法：getty [-h][-d<组态配置文件>][-r<延迟秒数>][-t<超时秒数>][-w<等待字符串>][终端机

编号][连线速率<终端机类型><管制线路>]

或：getty [-c<定义配置文件>]

使用说明：getty 指令是 UNIX 之类操作系统启动时所必需的 3 个步骤之一。

参数：

- -c<定义配置文件>：指定定义配置文件，预设为/etc/gettydefs。
- -d<组态配置文件>：指定组态配置文件，预设为/etc/conf.getty。
- -h：当传输速率为 0 时就强制断线。
- -r<延迟秒数>：设置延迟时间。
- -t<超时秒数>：设置等待登录的时间。
- -w<等待字符串>：设置等待回应的字符串。

## ifconfig

功能说明：显示或设置网络设备。

语法：ifconfig [网络设备][down up-allmulti-arp-promisc][add<地址>][del<地址>][<hw<网络设备类型><硬件地址>][io_addr<I/O 地址>][irq<IRQ 地址>][media<网络媒介类型>][mem_start<内存地址>][metric<数目>][mtu<字节>][netmask<子网掩码>][tunnel<地址>][-broadcast<地址>][-pointopoint<地址>][IP 地址]

使用说明：ifconfig 可设置网络设备的状态，或是显示目前的设置。

参数：

- add<地址>：设置网络设备 IPv6 的 IP 地址。
- del<地址>：删除网络设备 IPv6 的 IP 地址。
- down：关闭指定的网络设备。
- <hw<网络设备类型><硬件地址>：设置网络设备的类型与硬件地址。
- io_addr<I/O 地址>：设置网络设备的 I/O 地址。
- irq<IRQ 地址>：设置网络设备的 IRQ。
- media<网络媒介类型>：设置网络设备的媒介类型。
- mem_start<内存地址>：设置网络设备在主内存所占用的起始地址。
- metric<数目>：指定在计算数据包的转送次数时，所要加上的数目。
- mtu<字节>：设置网络设备的 MTU。
- netmask<子网掩码>：设置网络设备的子网掩码。
- tunnel<地址>：建立 IPv4 与 IPv6 之间的隧道通信地址。
- -broadcast<地址>：将要送往指定地址的数据包当成广播数据包来处理。
- -pointopoint<地址>：与指定地址的网络设备建立直接连线，此模式具有保密功能。
- -promisc：关闭或启动指定网络设备的 promiscuous 模式。
- IP 地址：指定网络设备的 IP 地址。

## netconf

功能说明：设置各项网络功能。

语法：netconf

使用说明：netconf 是 Red Hat Linux 发行版专门用来调整 Linux 各项设置的程序。

## netstat

功能说明：显示网络状态。

语法：netstat [-acCeFghilMnNoprstuvVwx][-A<网络类型>][--ip]

使用说明：利用 netstat 指令可让你得知整个 Linux 系统的网络情况。

参数：

- -a 或--all：显示所有连线中的 Socket。
- -A<网络类型>或--<网络类型>：列出该网络类型连线中的相关地址。
- -c 或--continuous：持续列出网络状态。
- -C 或--cache：显示路由器配置的块区信息。
- -e 或--extend：显示网络其他相关信息。
- -F 或--fib：显示 FIB。
- -g 或--groups：显示多重广播功能群组组员名单。
- -h 或--help：在线帮助。
- -i 或--interfaces：显示网络界面信息表单。
- -l 或--listening：显示监控中的服务器的 Socket。
- -M 或--masquerade：显示伪装的网络连线。
- -n 或--numeric：直接使用 IP 地址，而不通过域名服务器。
- -N 或--netlink 或--symbolic：显示网络硬件外围设备的符号连接名称。
- -o 或--timers：显示计时器。
- -p 或--programs：显示正在使用 Socket 的程序识别码和程序名称。
- -r 或--route：显示 Routing Table。
- -s 或--statistice：显示网络工作信息统计表。
- -t 或--tcp：显示 TCP 传输协议的连线状况。
- -u 或--udp：显示 UDP 传输协议的连线状况。
- -v 或--verbose：显示指令的执行过程。
- -V 或--version：显示版本信息。
- -w 或--raw：显示 RAW 传输协议的连线状况。
- -x 或--unix：此参数的效果和指定“-A unix”参数相同。
- --ip 或--inet：此参数的效果和指定“-A inet”参数相同。

## ping

功能说明：检测主机。

语法：ping [-dfnqrRv][-c<完成次数>][-i<间隔秒数>][-I<网络界面>][-l<前置载入>][-p<范本样式>][-s<数据包大小>][-t<存活数值>][主机名称或 IP 地址]

使用说明：执行 ping 指令会使用 ICMP 传输协议，发出要求回应的信息，若远端主机的网络功能没有问题，则会回应该信息，因而得知该主机运作正常。

参数：

- -d：使用 Socket 的 SO_DEBUG 功能。

- -c<完成次数>：设置完成要求回应的次数。
- -f：极限检测。
- -i<间隔秒数>：指定收发信息的间隔时间。
- -I<网络界面>：使用指定的网络界面送出数据包。
- -l<前置载入>：设置在送出要求信息之前，先行发出的数据包。
- -n：只输出数值。
- -p<范本样式>：设置填满数据包的范本样式。
- -q：不显示指令执行过程，开头和结尾的相关信息除外。
- -r：忽略普通的 Routing Table，直接将数据包送到远端主机上。
- -R：记录路由过程。
- -s<数据包大小>：设置数据包的大小。
- -t<存活数值>：设置存活数值 TTL 的大小。
- -v：详细显示指令的执行过程。

# 附录 B

# 网络工具资源汇总



### 1．Openbsd PF：OpenBSD 数据包过滤器

OpenBSD

像其他平台上的 Netfilter 和 IP Filter 一样，OpenBSD 用户最爱用 PF，这就是他们的防火墙工具。它的功能有网络地址转换、管理 TCP/IP 通信、提供带宽控制和数据包分级控制。它还有一些额外的功能，例如被动操作系统检测。PF 是由编写 OpenBSD 的同一批人编写的，所以你完全可以放心使用，它已经经过了很好的评估、设计和编码以避免暴露其他包过滤器（other packet filters）上的类似漏洞。

### 2．Arpwatch：ARP 中间人攻击检测器（Linux）



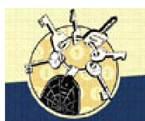
Arpwatch 是 LBNL 网络研究组出品的一款经典的 ARP 中间人（man-in-the-middle）攻击检测器。它记录网络活动的系统日志，并将特定的变更通过 E-mail 报告给管理员。Arpwatch 使用 LibPcap 来监听本地以太网接口 ARP 数据包。

### 3．OSSEC HIDS：开源的基于主机的入侵检测系统




OSSEC HIDS 的主要功能有日志分析、完整性检查、Rootkit 检测、基于时间的报警和主动响应。除了具有入侵检测系统功能外，它一般还被用在 SEM/（Security Event Management，安全事件管理），SIM/（Security Information Management，安全信息管理）解决方案中。因其强大的日志分析引擎，ISP（Internet Service Provider，网络服务提供商）、大学和数据中心用其监控和分析他们的防火墙、入侵检测系统、网页服务和验证等产生的日志。

### 4．GnuPG/PGP ：对您的文件和通信进行高级加密



PGP 是 Phil Zimmerman 出品的著名加密程序，可以使您的数据免受窃听以及其他危险。GnuPG 是一款口碑很好的遵守 PGP 标准的开源应用（可执行程序名为 gpg）。GunPG 是免费的，而 PGP 对某些用户是收费的。

### 5．Retina：eEye 出品的商业漏洞评估扫描器




像 Nessus 一样，Retina 的功能是扫描网络中所有的主机并报告发现的所有漏洞，由 eEye 出品，此公司以其 security research 而闻名。

### 6．Sysinternals：一款非常全面的 Windows 工具集合



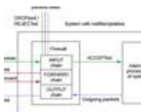

Sysinternals 为 Windows 低级入侵提供很多非常有用的小工具。其中一部分是免费的，有些还附有源代码，其他是需要付费使用的。受访者最喜欢此集合中的工具，如：ProcessExplorer 监视所有进程打开的所有文件和目录（类似 UNIX 上的 LSoF），PsTools 管理（执行、挂起、杀死、查看）本地和远程进程，Autoruns 发现系统启动和登录时加载了哪些可执行程序，RootkitRevealer 检测注册表和文件系统 API 异常，以及用于发现用户模式或内核模式的 Rootkit 工具。

### 7．GFI Languard：Windows 平台上的商业网络安全扫描器




GFI 公司的 Languard 扫描 IP 网络，以检测哪些机器正在运行的程序。如收集 Windows 机器的 Service Pack 级别、错失的安全补丁、无线接入点、USB 设备、开放式股票、开放口岸、活跃于电脑中的服务/应用关键注册表项、弱密码、用户和用户组，以及更多。扫描结果保存为一个 HTML 报告，其中可以定制/质疑。它也包括一个补丁管理器，可侦测并安装失踪补丁。免费试用版本可用，但只有 30 天之久。

### 8．Netfilter：最新的 Linux 核心数据包过滤器防火墙



Netfilter 是最新的 2.4.x 版本 Linux 内核集成的 IP 信息包过滤系统。如果 Linux 系统连接到互联网或 LAN、服务器或连接 LAN 和互联网的代理服务器，则该系统有利于在 Linux 系统上更好地控制 IP 信息包过滤和防火墙配置。Mugdha Vairagade 将介绍 Netfilter/Iptables 系统，它是如何工作的，它的优点、安装和配置，以及如何使用它来配置 Linux 系统上的防火墙以过滤 IP 信息包。

### 9．Ettercap：为交换式局域网提供更多保护




Ettercap 是一款基于终端的以太网局域网嗅探器/拦截器/日志器。它支持主动和被动的多种协议解析（甚至是 ssh 和 https 加密过的），还可以进行已建立连接的数据注入和实时过滤，保持连接同步。大部分嗅探模式都是强大且全面的嗅探组合，支持插件，能够识别您是否处在交换式局域网中，通过使用操作系统指纹（主动或被动）技术可以得到局域网结构。

10．Snort：一款广受欢迎的开源 IDS（入侵检测系统）工具



这款小型的入侵检测和预防系统擅长于通信分析和 IP 数据包登录（packet logging）。Snort 除了能够进行协议分析、内容搜索和包含其他许多预处理程序，还可以检测上千种蠕虫病毒、漏洞、端口扫描以及其他可疑行为。Snort 使用一种简单的基于规则的语言来描述网络通信，以及判断对于网络数据是放行还是拦截，其检测引擎是模块化的。用于分析 Snort 警报的网页形式的引擎 Basic Analysis and Security Engine（BASE）可免费获得。开源的 Snort 为个人、小企业、集团用户提供良好的服务。其母公司 SourceFire 提供丰富的企业级特性和定期升级，以丰富其产品线，提供（必须注册）5 天免费的规则试用。您也可以在 Blee** Edge Snort 找到很多免费规则，点击下载。

11．X-scan：一款网络漏洞扫描器



X-Scan 是一款多线程、支持插件的漏洞扫描器，主要功能包括全面支持 NASL（Nessus Attack Scripting Language，Nessus 攻击脚本语言）、检测服务类型、远程操作系统类型（版本）检测、弱用户名/密码匹配等，点击下载。

12．Spike Proxy：HTTP 攻击检测




Spike Proxy 是一款开源的以发现网站漏洞为目的的 HTTP 代理。它是 Spike Application Testing Suite 的一部分，功能包括自动 SQL 注入检测、网站爬行（Web site crawling）、登录列表暴力破解、溢出检测和目录游走检测。

13．Yersinia：支持多协议的底层攻击检测工具



Yersinia 是一款底层协议攻击入侵检测工具，它能实施针对多种协议的多种攻击，例如夺取生成树的根角色（生成树协议，Spanning Tree Protocol）、生成虚拟 CDP（Cisco Discovery Protocol，Cisco 发现协议）邻居、在一个 HSRP（Hot Standby Router Protocol，热等待路由协议）环境中虚拟成一个活动的路由器、制造假 DHCP 反馈，以及其他底层的攻击。

14．chkrootkit：本地 Rootkit 检测器




chkrootkit 是一款小巧易用的 UNIX/Linux 平台上的可以检测多种 Rootkit 入侵的工具。它的功能包括检测文件修改、utmp/wtmp/last 日志修改、界面欺骗（promiscuous interfaces）、恶意核心模块（malicious kernel modules）。

15．rkhunter：一款 UNIX 平台上的 Rootkit 检测器




rkhunter 是 Linux 下的一款开源入侵检测工具。rkhunter 具有比 chrootkit 更为全面的扫描范围。除 Rootkit 特征码扫描外，rkhunter 还支持端口扫描、常用开源软件版本和文件变动情况检查等。

16．Angry IP Scanner：Windows IP 扫描器和端口扫描器



Angry IP Scanner 是一个相当小的 IP 扫描软件，可以在最短的时间内扫描远端主机 IP 的运作状况，并且快速地将结果整理完汇报给您知晓。Angry IP Scanner 可以扫描的项目很多，包括远端主机的名称、目前开启的通联埠以及 IP 的运作状况等，让您可以完全掌握对方主机的运作状况，对于网管人员来说，实在是个不可多得的好帮手。

17．RainbowCrack：极具创新性的密码哈希破解器



RainbowCrack 是一个使用内存时间交换技术（Time-Memory Trade-Off Technique）加速口令破解过程的口令破解器。RainbowCrack 使用了彩虹表，也就是一张预先计算好的明文和散列值的对照表。通过预先花费时间创建这样的彩虹表，能够在以后破解口令时节约大量的时间。

18．Pwdump：一款 Windows 密码破解和恢复工具



Pwdump 可以从 Windows 主机中取得 NTLM 和 LanMan 哈希值，无论系统密码是否启用，它还能显示系统中存在的历史密码。数据输出格式为 L0phtcrack 兼容格式，也可以以文件形式输出数据。

19．IDA Pro：Windows 或 Linux 反编译器和调试器



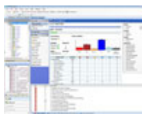
IDA Pro Advanced 是目前最棒的一个静态反编译软件，是破解者不可缺少的利器！巨酷的反编译软件，破解高手们几乎都喜欢用这个软件。

20．Xprobe2：主动操作系统指纹工具

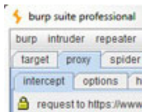


Xprobe 是一款远程主机操作系统探查工具。开发者是基于和 Nmap 相同的一些技术并加入了自己的创新研制而成的。Xprobe 通过 ICMP 协议来获得指纹。

21．WebInspect：强大的网页程序扫描器


SPI Dynamics’ WebInspect 应用程序安全评估工具，帮你识别已知和未知的网页层漏洞。它还能检测 Web 服务器的配置属性，以及进行常见的网页攻击，例如参数注入、跨网站脚本、目录游走等。

22．Burp Suite：Web 应用程序攻击集成平台



Burp Suite 是一个 Web 应用程序集成攻击平台，它包含了一系列 Burp 工具，这些工具之间有大量接口可以互相通信，之所以这样设计是为了促进和提高整个攻击的效率。平台中所有工具共享同一 robust 框架，以便统一处理 HTTP 请求、持久性、认证、上游代理、日志记录、报警和可扩展性。Burp Suite 允许攻击者结合手工和自动技术去枚举、分析、攻击 Web 应用程序。这些不同的 Burp 工具通过协同工作，有效地分享信息，支持以某种工具的信息为基础供另一种工具使用的方式发起攻击。

23．NBTscan：在 Windows 网络上收集 NetBIOS 信息


NBTscan 是一款在 IP 网络上扫描 NetBIOS 名称信息的工具。它通过给指定范围内所有地址发送状态查询来获得反馈信息并以表的形式呈现给使用者。每一地址的反馈信息包括 IP 地址、NetBIOS 计算机名、登录用户、MAC 地址。

## 24．P0f：新的远程 OS 指纹被动判别工具



P0f 能够通过捕获并分析目标主机发出的数据包来对主机上的操作系统进行鉴别，即使是在系统上装有性能良好的防火墙的情况下也没有问题。P0f 不增加任何直接或间接的网络负载，没有名称搜索、没有秘密探测、没有 ARIN 查询，什么都没有。某些高手还可以用 P0f 检测出主机上是否有防火墙存在、是否有 NAT、是否存在负载平衡器，等等。

## 25．AirSnort：802.11 WEP 加密破解工具



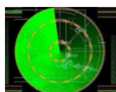

AirSnort 是一款用来恢复加密密码的无线 LAN（WLAN）工具，由 Shmoo Group 出品。其工作原理是被动监控传输信息，当收集到足够多的数据包后开始计算加密密码。

## 26．L0phtCrack：Windows 密码猜测和恢复程序



L0phtCrack 也叫做 LC5，用来尝试通过哈希（通过某种访问方式获得的）方法破解诸如 Windows NT/2000 工作站、联网服务器、主域控制器，或活动目录密码，有时它也可以通过嗅探获得密码的哈希值。它还可以通过多种手段来猜测密码（字典、暴力破解等）。Symantec 公司 2006 年已经停止了 LC5 的开发，但 LC5 installer 的安装文件随处可以找到。免费试用版只能使用 15 天，Symantec 已经停止出售此软件的注册码，所以如果您不想放弃使用它，就必须找到一个与其对应的注册码生成器（key generator）。因为 Symantec 不再维护此软件，所以最好尝试用 Cain and Abel 或 John the Ripper 来代替之。

## 27．Aircrack：最快的 WEP/WPA 破解工具



Aircrack 是一套用于破解 8WEP 和 WPA 的工具套装，一般用于无线网络的密钥破解，从而非法进入未经许可的无线网络。只要一旦收集到足够的加密数据包，利用它就可以破解 40～512 位的 WEP 密匙，也可以通过高级加密方法或暴力破解来破解 WPA1 或 WPA2 网络。

## 28．SuperScan：只运行于 Windows 平台之上的端口扫描器



SuperScan 是一款 Foundstone 公司开发的免费的只运行于 Windows 平台之上的不开源的 TCP/UDP 端口扫描器，其中还包含许多其他网络工具，例如 ping、路由跟踪、http head 和 whois。

### 29．NetStumbler：免费的 Windows 802.11 嗅探器



NetStumbler 是广为人知的寻找开放无线访问接入点的 Windows 工具（wardriving）。其 PDA 上的 Win CE 系统版本名叫 Ministumbler。此软件当前免费，但只能够运行在 Windows 平台上，且代码不公开。它使用很多主动方法寻找 WAP，而 KisMET 或 KisMAC 则更多地使用被动嗅探。

### 30．Dsniff：一款超强的网络评估和渗透检测工具套装


由 Dug Song 精心设计并广受欢迎的这款套装包含很多工具，Dsniff、filesnarf、mailsnarf、msgsnarf、urlsnarf 和 webspy 通过被动监视网络以获得敏感数据（例如密码、邮件地址、文件等）。Arpspoof、dnsspoof 和 macof 能够拦截一般很难获取到的网络通信信息（例如由于使用了第二层转换（layer-2 switching））。Sshmitm 和 webmitm 通过 ad-hoc PKI 中弱绑定漏洞对 ssh 和 https 会话进行重定向，实施动态 monkey-in-the-middle（利用中间人攻击技术，对会话进行劫持）攻击。Windows 版本可以在这里获取。总之，这是一个非常有用的工具集，它能完成几乎所有密码嗅探需要做的工作。

### 31．THC Hydra：支持多种服务的最快的网络认证破解器


如果您需要暴力破解一个远程认证服务，Hydra 经常会是选择对象。它可以同时对 30 个以上的端口进行基于字典的快速破解，包括 Telnet、FTP、HTTP、HTTPS、SMP、多种数据库及其他服务。和 THC Amap 一样，此 Hydra 版本来自于民间组织 THC。

### 32．Paros proxy：网页程序漏洞评估代理



Paros proxy 是基于 Java 的网页程序漏洞评估代理，支持实时编辑和浏览 HTTP/HTTPS 信息，例如修改 Cookie 和表字段中的内容。它包含网页通信记录器、网页小偷（Web spider）、哈希计算器和一个常用网页程序攻击扫描器，如 SQL 注入和跨网站脚本等。

### 33．John the Ripper：一款强大简单的支持多平台的密码破解器


John the Ripper 是一款最快的密码破解器，当前支持多种主流 UNIX（官方支持

11 种，没有计算不同的架构）、DOS、Windows 32、BeO 和 OpenVMS。它的主要功能就是检测弱 UNIX 密码。它支持主流 UNIX 下的多种（3 种）哈希加密类型，它们是 Kerberos、AFS 以及 Windows NT/2000/XP LM。其他哈希类型可以通过补丁包加载。如果您希望从一些单词表开始的话，您可以在这里找到。

### 34．Cain & Abel：Windows 平台上最好的密码恢复工具


UNIX 用户经常声称正因为 UNIX 平台下有很多非常好的免费安全工具，所以 UNIX 才会成为最好的平台，而 Windows 平台一般不在他们的考虑范围之内。他们也许是对的，但 Cain & Abel 确实让人眼前一亮。这种只运行于 Windows 平台的密码恢复工具可以做很多事情。它可以通过嗅探网络来找到密码、利用字典破解加密密码、暴力破解密码和密码分析、记录 VoIP 会话、解码非常复杂的密码、星号查看、剥离缓存密码以及分析路由协议。另外，其文档也很齐全（well documented）。

### 35．Nessus：最好的 UNIX 漏洞扫描工具



Nessus 是最好的免费网络漏洞扫描器，它可以运行于几乎所有的 UNIX 平台之上。它不仅可以永久升级，还免费提供多达 11 000 种插件（但需要注册并接受 EULA-acceptance——终端用户授权协议）。它的主要功能是远程或本地（已授权的）安全检查，客户端/服务器架构，GTK（Linux 下的一种图形界面）图形界面，内置脚本语言编译器，可以用其编写自定义插件，或用来阅读别人写的插件。Nessus 3 已经开发完成（now closed source），其现阶段仍然免费，而您想获得最新的插件就要收费。

### 36．Wireshark：网络嗅探工具



Wireshark（2006 年夏天之前叫做 Ethereal）是一款非常棒的 UNIX 和 Windows 上开源网络协议分析器。它可以实时检测网络通信数据，也可以检测其抓取的网络通信数据快照文件。通过图形界面浏览这些数据，查看网络通信数据包中每一层的详细内容。Wireshark 拥有许多强大的特性：包含有强显示过滤器语言（rich display filter language）和查看 TCP 会话重构流的能力；支持上百种协议和媒体类型；拥有一个类似 tcpdump（一个 Linux 下的网络协议分析工具）的名为 ethereal 的命令行版本。不得不说一句，Ethereal 已经饱受许多可远程利用的漏洞折磨，所以要经常对其进行升级，并在不安全网络或敌方网络（例如安全会议的网络）中谨慎使用。

### 37．Netcat：网络瑞士军刀



这个简单的小工具可以读/写经过 TCP 或 UDP 网络连接的数据。它被设计成一个可靠的可以被其他程序或脚本直接和简单使用的后台工具。同时，它也是一个功能多样的网络调试和检查工具，因为它可以生成几乎所有您想要的网络连接，包括通过端口绑定来接受输入连接。Netcat 最早由 Hobbit 在 1995 年发布，但在其广为流传的情况下并没有得到很好的维护。现在 nc110.tgz 已经很难找了。这个简单易用的工具使很多人写出了很多其他的 Netcat 应用，有很多功能都是原版本没有的，其中最有趣的是 Socat，它将 Netcat 扩展成可以支持多种其他 socket 类型、SSL 加密、SOCKS 代理，以及其他扩展的更强大的工具。还有 Chris Gibson’s Ncat，能够提供更多对便携设备的支持。其他基于 Netcat 的工具还有 OpenBSD’s nc、Cryptcat、Netcat6，Pnetcat 和 SBD。Netcat 又叫做 GNU Netcat。

### 38．Metasploit Framework：黑掉整个星球



2004 年 Metasploit 的发布在安全界引发了强烈的地震，超过了很多广为流传的诞生了几十年的老牌工具。它是一个强大的开源平台，供开发、测试和使用恶意代码。这种可扩展的模型将负载控制、编码器、无操作生成器和漏洞整合在一起，使得 Metasploit Framework 成为一种研究高危漏洞的途径。它自带上百种漏洞，还可以在 online exploit buil** demo（在线漏洞生成演示）看到如何生成漏洞。这使得您自己编写漏洞变得更简单，它势必将提升非法 Shellcode 代码的水平，扩大网络阴暗面。与其相似的专业漏洞工具，例如 Core Impact 和 Canvas 已经被许多专业领域用户使用。Metasploit 降低了这种能力的门槛，将其推广给大众。

### 39．Hping2：一种网络探测工具，是 ping 的超级变种




这个小工具可以发送自定义的 ICMP、UDP 和 TCP 数据包，并接收所有反馈信息。它的灵感来源于 ping 命令，但其功能远远超过 ping。它还包含一个小型的路由跟踪模块，并支持 IP 分段。此工具可以在常用工具无法对有防火墙保护的主机进行路由跟踪/ping/探测时大显身手。它经常可以帮助您找出防火墙的规则集，当然还可以通过它来学习 TCP/IP 协议，并做一些 IP 协议的实验。

### 40．Kismet：一款超强的无线嗅探器




Kismet 是一款基于命令行（ncurses）的 802.11 layer2 无线网络探测器、嗅探器和入侵检测系统。它对网络进行被动嗅探（相对于许多主动工具，例如 NetStumbler），可以发

现隐形网络（非信标）。它可以通过嗅探 TCP、UDP、ARP 和 DHCP 数据包来自动检测网络 IP 段，以 Wireshark/TCPDump 兼容格式记录通信日志，更加可以将被检测到的网络分块并按照下载的分布图进行范围估计。如您所想，这款工具一般被 wardriving 所使用。嗯！还有 warwalking、warflying 和 warskating…

### 41．Nagios：一款开源的主机、服务和网络监控程序




Nagios 是一款系统和网络监控程序。它可以监视您指定的主机和服务。当被监视对象发生任何问题或问题被解决时，发出提示信息。它的主要功能有监控网络服务（SMTP、POP3、HTTP、NNTP、Ping 等）、监控主机资源（进程负载、硬盘空间使用情况等）、当发现问题或问题解决时通过多种形式发出提示信息（E-mail、寻呼机或其他用户定义的方式）。

### 42．Fport：Foundstone 出品的加强版 netstat




Fport 可以报告所有本地机上打开的 TCP/IP 和 UDP 端口，并显示是何程序打开的端口。所以用它可以快速地识别出未知的开放端口以及与其相关的应用程序。它只有 Windows 版本，但现在很多 UNIX 系统上的 netstat 也提供同样的功能（Linux 请用 netstat -pan）。SANS 的文章有 Fport 的使用说明和结果分析方法。

### 43．Tor：匿名网络通信系统




Tor 是一款面向希望提高其网络安全性的广大组织和大众的工具集。Tor 有匿名网页浏览和发布、即时信息、irc、ssh 以及其他一些 TCP 协议相关的功能。Tor 还为软件开发者提供一个可开发内置匿名性、安全性和其他私密化特性的软件平台。在 Vidalia 可以获得跨平台的图形化界面。

### 44．Firewalk：高级路由跟踪工具（Linux）




Firewalk 使用类似 traceroute 的技术来分析 IP 包的响应，从而测定网关的访问控制列表和绘制网络图。Firewalk 使用类似路由跟踪（traceroute-like）的 IP 数据包分析方法，来测定一个特殊的数据包是否能够从攻击者的主机传送到位于数据包过滤设备后的目标主机。

### 45．EtherApe：UNIX 平台上的图形界面网络监控器




EtherApe 包含连接层、IP 和 TCP 三种模式，其网络活动图通过不同颜色来标识不同协议，主机和连接的图形大小随通信情况而变化。它支持以太网、FDDI、令牌环、ISDN、PPP 和 SLIP 设备。它可以实施过滤网络通信，也可以抓取网络通信快照文件。

### 46．OpenSSL：最好的 SSL TLS 加密库




OpenSSL 项目的目的是通过开源合作精神开发一种健壮的、可以和同类型商业程序媲美的、全功能的，且开源的应用于 SSL v2/v3（Secure Sockets Layer）和 TLS v1（Transport Layer Security）协议的普遍适用的加密库工具集。本项目由世界范围内的志愿者维护，他们通过互联网联络，计划和开发 OpenSSL 工具集及其相关文档。

### 47．Ngrep：方便的数据包匹配和显示工具




Ngrep 是 grep（在文本中搜索字符串的工具）的网络版，它力求更多的 grep 特征，用于搜寻指定的数据包。正由于安装 Ngrep 需用到 libpcap 库，所以支持大量的操作系统和网络协议，能识别 TCP、UDP 和 ICMP 包，理解 bpf 的过滤机制。

### 48．Ntop：网络通信监控器




Ntop 以类似进程管理器的方式显示网络使用情况。在应用程序模式下，它能显示用户终端的网络状况。在网页模式下，它作为网页服务器，以 HTML 文档形式显示网络状况。它是 NetFlow/sFlow 发射和收集器，通过一个基于 HTTP 的客户端界面来生成以 Ntop 为中心的监控程序，RRD（Round Robin Database，环形数据库）用来持续储存网络通信状态信息。

### 49．Tripwire：一款老牌的文件完整性检查器




Tripwire 是一款最为常用的开放源码的完整性检查工具，它生成目标文件的校验和并周期性地检查文件是否被更改。例如，如果 Tripwire 已经为/bin/login 和 /bin/ls 存放了快照，那么对它们的尺寸、inode 号、权限以及其他属性的任何修改，都逃不过 Tripwire 的火眼金睛。尤其是对于文件内容的修改，即使只改变了一个字节，Tripwire 也能察觉得到，因为校验和是针对文件整体的。

50．WebScarab：分析 HTTP 和 HTTPS 协议的应用程序框架

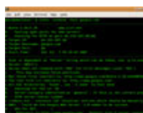

WebScarab 的原理很简单，它记录检测到的会话内容（请求和应答），使用者可以通过多种形式来查看记录。WebScarab 的设计目的是让使用者可以掌握某种基于 HTTP（S）程序的运作过程；也可以用它来调试程序中较难处理的 bug，还可以帮助安全专家发现潜在的程序漏洞。

51．Scapy：交互式数据包处理工具



Scapy 是一款强大的交互式数据包处理工具、数据包生成器、网络扫描器、网络发现工具和包嗅探工具。它提供多种类别的交互式生成数据包或数据包集合、对数据包进行操作、发送数据包、包嗅探、应答和反馈匹配等功能。Python 解释器提供交互功能，所以要用到 Python 编程知识（例如 variables、loops 和 functions）。Scapy 支持生成报告，且报告生成简单。

52．Nikto：非常全面的网页扫描器


Nikto 是一款开源的 GPL 网页服务器扫描器，它可以对网页服务器进行全面的多种扫描，包含超过 3200 种有潜在危险的文件/CGIs；超过 625 种服务器版本；超过 230 种特定服务器问题。扫描项和插件可以自动更新（如果需要）。它基于 Whisker/libwhisker 完成其底层功能，是一款非常棒的工具；但其软件本身并不经常更新，最新和最危险的可能检测不到。

53．Safe3 Web 应用防火墙


Safe3 Web 应用防火墙是国内知名安全组织，保护伞网络，基于新一代 Web 安全技术开发的全方位企业 Web 信息安全产品。它不仅能有效地扫描各种 WebShell，而且可以抵御各种 Web 攻击。

54．Canvas：一款全面的漏洞检测框架



Canvas 是 Aitel’s ImmunitySec 出品的一款漏洞检测工具，可检测 150 个以上的漏洞。它比 Core Impact 便宜一些，但是价值也达数千美元。您也可以购买 VisualSploit Plugin 实现在图形界面上通过拖曳就可以检测漏洞。Canvas 偶尔也会发现一些 0Day 漏洞。

奠基
计算机网络
修订版
韩立刚 王艳华 潘刚柱 张庆力 编著
- 计算机网络基本功不扎实？
- 培训不能费，就是学不会？
- 经验只能靠时间来积累？
清华大学出版社

构建虚拟专用通道
OpenVPN服务器详解与架设指南
（基于Linux）
清华大学出版社

掌控
构建Linux系统
Nagios监控服务器
陶利军 编著
清华大学出版社

# 防线

## 企业Linux安全运维理念和实战

“开卷有益，向行业领先者学习安全理念、策略和技术”
“工欲善其事，必先利其器！使用开源技术和工具构筑企业信息安全防线”

本书是一本面向企业的基于操作系统平台进行信息安全建设的实践书籍，是围绕着“什么是”、“为什么”和“怎么样”构建信息安全体系。

本书并不需要读者具有高深的计算机科学技术或者信息安全的基础理论知识，主要面向怀有一定的工作目标并对信息安全有一定兴趣的工程师或者信息安全从业人员。

笔者也非常希望企业高层（包括CIO、CEO和CSO们）能够从本书中获得他们需要的一些宝贵理念和实践指引。本书中的一些理念、方法和实践指南，在实际企业信息安全建设中与一些CIO、CEO们经过讨论和达成共识。

刚走出校门的大学生、研究生以及正在通过各种渠道（包括培训、实习等）来试图进入信息安全领域的学子或者工作者，相信可以从本书中获得系统知识和工作指南。本书所阐述的知识、操作所用的案例都是在实际工作中精挑细选的，相信你们可以从中提前感受和体会到作为一名信息安全从业人员所需要具备的基本知识结构以及实在的技能。

“棱镜”暴露了什么问题？安全！在此向各位朋友推荐这本书，也祝愿李先生的作品能够让更多读者从中得到收获。

**——51CTO网站副总编 赵磊**

如果您正在为公司的安全问题担忧，不妨考虑阅读这本书，希望读者在遇到问题时，通过这本书可以“众里寻TA千百度，蓦然回首，答案只在桌上手边处”。

**——汇美国际下属科技公司CTO 战剑新**

Doctor Yang Li has been a senior expert with extensive experience in network security. This book is his another masterpiece in security field which is almost rated as the perfect combination of security theory and practice, to some extent, it is a bible book of security.
WORTHWHILE READING!!!

**——Andrew Liu, the CTO of Mobile Payment Inc. @Silicon Valley**

51CTO.com 技术成就梦想 真心推荐

关于本书的任何问题，可以到作者博客上寻求答案或联系作者，必将百问不厌：
http://patterson.blog.51cto.com/


清华大学出版社数字出版网站
WQBook 书文局泉
www.wqbook.com



ISBN 978-7-302-31807-1





定价：79.00元

# 看完了

如果您对本书内容有疑问，可发邮件至contact@turingbook.com，会有编辑或作译者协助答疑。也可访问图灵社区，参与本书讨论。

如果是有关电子书的建议或问题，请联系专用客服邮箱：ebook@turingbook.com。

在这里可以找到我们：

微博 @图灵教育 : 好书、活动每日播报
微博 @图灵社区 : 电子书和好文章的消息
微博 @图灵新知 : 图灵教育的科普小组
微信 图灵访谈 : ituring_interview，讲述码农精彩人生
微信 图灵教育 : turingbooks